Learning Terminological Knowledge with High Confidence from Erroneous Data

D I S S E R T A T I O N

zur Erlangung des akademischen Grades

Doctor rerum naturalium
(Dr. rer. nat.)

vorgelegt

der Fakultät Mathematik und Naturwissenschaften
der Technischen Universität Dresden

von

Dipl.-Math. Daniel Borchmann

geboren am 23. November 1984 in Königs Wusterhausen

Eingereicht am 23. Mai 2014
Verteidigt am 9. September 2014

Die Dissertation wurde in der Zeit von September 2009 bis
Mai 2014 im Institut für Algebra angefertigt.

Gutachter

Prof. Dr. Bernhard Ganter, Technische Universität Dresden
Prof. Dr. Sergei O. Kuznetsov, Higher School of Economics

„Wenn ich nur erst die Sätze habe! Die Beweise werde ich schon finden.“

“If only I had the theorems! Then I should find the proofs easily enough.”

——— Bernhard Riemann

Preface

Knowledge is necessary to solve problems and to act intelligently in almost all situations – a simple fact that is true for humans and computers alike. Therefore, to enable humans and computers to act intelligently, it is necessary to first solve the problem of *learning relevant knowledge*, more formally referred to as *knowledge acquisition*. In the case of computers, another problem has to be tackled, namely the problem of *representing knowledge* in a way that is suitable for a machine consumption.

A common approach to representing knowledge suitable for computer consumption is to use different kinds of logics, and among these *description logics* are a very popular choice. Description logics are a family of logic-based formalisms with varying reasoning complexity and expressiveness, specifically tailored towards practical decision procedures. Moreover, *description logic knowledge bases* (also called *ontologies*) allow for a powerful mechanism to represent knowledge, which is used in practical applications like bio-medical ontologies and the semantic web.

The types of knowledge representable by description logic knowledge bases are *assertional knowledge* and *terminological knowledge*. Obtaining assertional knowledge, like the fact that Abraham Lincoln was a president, or that John McCarthy was a professor at Stanford University, is comparably easy, and can (to a certain extent) even be done automatically. This is mostly due to the rather local nature of assertional knowledge, which just concerns one or two individuals. On the other hand, obtaining terminological knowledge is much more involved, because this knowledge relates concepts, and not individuals, to each other: the fact that every human is mortal, or that every human has a head are examples for terminological knowledge. Obtaining such knowledge, which may concern a wide and mainly indefinite range of individuals or things, is much harder, and hoping to obtain it in an automatic way seems gullible.

Nevertheless, there have been various approaches to obtaining terminological knowledge, at least in a preliminary version, from various sources of data, like natural language texts, databases, or linked data. One of these approaches has been developed by Franz Baader and Felix Distel, and is based on notions from the mathematical theory of *formal concept analysis*. This approach allows the extraction of general terminological knowledge in the form of *general concept inclusions* that are valid in a given *finite interpretation*. Interpretations are structures, which are used in description logics to define the semantics of different

logics, and can be thought of as directed edge- and vertex-labeled graphs. Because of their graph-like nature, interpretations can be thought of as a variant of *linked data*, and then the approach by Baader and Distel in principle allows the extraction of terminological knowledge from this form of data, which is ubiquitous in the realm of the semantic web. Thus, one could turn the vast amount of linked data into description logic knowledge bases and use them to make computers act more intelligently.

However, the approach of Baader and Distel has some drawbacks which make it hard to apply it to real-world data. One of these is the fact that every real-world data set contains *errors*, and in such cases considering only terminological knowledge that is *valid* in this erroneous data seems futile. In the worst case, valid terminological knowledge is not extracted from the data set because a single error invalidates it.

On the other hand, learning terminological knowledge from an interpretation or linked data is only reasonable if this data set is of sufficiently high *quality*, meaning that it does not contain too many errors that are relevant for the learning process. Then one could imagine that it may be fruitful to also consider terminological knowledge that is *almost valid* in this data. In this way, knowledge that is erroneously invalidated by few errors is still recovered from the data. Of course, one has to make sure that rare but valid counterexamples in this data set are not accidentally ignored in this way, i. e. the knowledge obtained needs to be verified by an external source of information, like a human expert.

It is the purpose of this work to present an extension of the results by Baader and Distel that incorporates the idea of considering general concept inclusions which are almost valid in the input data. To this end, we shall make use of the notion of *confidence* from data mining, and transfer it to general concept inclusions. We then shall show how we can learn all general concept inclusions with *high confidence* in our input data. For this we use ideas from formal concept analysis, more specifically results obtained by Michael Luxenburger on *partial implications* in formal contexts. We then shall apply our findings to a small example interpretation extracted from the DBpedia data set, and assess in how far our approach yields more error-tolerant results. Finally, we shall show how we can obtain a semi-automatic algorithm for the expert-based verification step mentioned above, based on the algorithm of *attribute exploration* from formal concept analysis.

All these extensions are based on the original results by Baader and Distel, and we shall review these results to the extent needed for our considerations. This will also include an introduction to the main notions of formal concept analysis and description logics, as necessary for this work. It is the aspiration that this extension of Baader's and Distel's results make the techniques of extracting general concept inclusions from finite interpretations more accessible for practical applications.

Acknowledgments As any work of this size, this thesis would not have been possible without the (explicit or implicit) help of many people. First and foremost, I would like to express my thanks to my supervisors, Prof. Bernhard Ganter, Prof. Franz Baader, and Prof. Gerd Brewka, for their encouragement and help during the development of this dissertation, as

well as for their financial support. I am particularly grateful for the opportunity of being a scholarship holder of the Research Training Group 1763 “QuantLA” during the last two years, which gave me the necessary time and confidence to work out what has previously been only a loose collection of vague ideas.

One of the biggest obstacles to finishing this project certainly was to sustain the motivation for doing so, and I owe special thanks to a lot of people for helping me in this, some of them maybe without knowing it. In particular, I am grateful to all members of the Institute of Theoretical Computer Science, as well as to all members of the Institute of Algebra, for fruitful discussions ranging from technical topics up to seemingly unrelated things, that however helped me to forget about this work for a while, giving me the necessary pause. This especially includes Felix Distel, whom I had the honor to work with for nearly four years, and on whose Ph. D. thesis the present book is based. I am also grateful to the many anonymous reviewers who thoroughly pointed out errors in my papers, and thus helped to improve the quality of my results. Finally, I have to thank Prof. Ulrike Baumann for sharing with me her wisdom about life, which helped to overcome more than one crisis.

This book is typeset with LaTeX, the only professional and free typesetting system I am aware of that is able to produce typographic results of high (and thus acceptable) quality. Using LaTeX together with Emacs on Debian GNU/Linux gave me just the right tools to master the complexities a book of over 200 pages brings with it, and even allowed me to have fun while writing it. Thanks to all who contributed to these tools, and who contribute to free software in general!

What I am particularly bad at is proof-reading, and thus I am deeply indebted to my two personal lectors, Juliane Prochaska and Tom Hanika, for carefully reading *the whole book* and giving a lot of helpful hints. Without them, many sentences would still be too long to be understandable.

One of the deepest insights I have gained during the past years is that there is much more to life than working, and there is no deeper, more profound, and more lasting experience than receiving this insight from ones own children. Thank you, Tabea and Fiona, for showing me the right way. And thank you, Juliane, for our wonderful children. Thank you!

Contents

Chapter 1

Introduction

For a machine to interact in an intelligent manner with its outside world, it needs to be equipped with suitable knowledge. The very same is true for humans, and acquiring this knowledge is a highly complex task which up to today is not completely understood. To help within this process, previously acquired knowledge is *represented* in different ways, e. g. as books or films, to help others learning it.

However, the way knowledge is represented for human consumption is almost always not suitable for machines. While humans can extract knowledge from natural language, which may be full of ambiguity, machines require precise formulations of knowledge. The representation of knowledge in a machine-understandable way is the main focus of the area of *knowledge representation* [56].

One of the most successful approaches to knowledge representations are description logics [12], a family of logic-based knowledge representation formalisms with varying expressivity and reasoning complexity. Description logics allow to represent knowledge as *knowledge bases*, or *ontologies*. Essentially, these are collections of axioms, which may be either *assertional* or *terminological*. For example, to represent the fact (the *assertion*) that an individual tom is a cat, one can use the assertional axiom

$$\mathsf{Cat}(\mathsf{tom}). \tag{1.1}$$

On the other hand, to state the terminological knowledge that every cat hunts mice, one can use so-called *general concept inclusions* (GCIs) and write

$$\mathsf{Cat} \sqsubseteq \exists \mathsf{hunts}.\mathsf{Mouse}. \tag{1.2}$$

As soon as a knowledge base is available, it is possible to *reason* with it, i. e. to extract knowledge from it that is implicitly contained. For example, from the two axioms stated above, we can infer that the individual tom hunts mice, although this has never been stated explicitly.

Knowledge bases can be used to represent knowledge in a machine-consumable way. However, the question arises how such knowledge bases can be obtained. One way or

the other, knowledge which is available to humans has to be translated into the form of a description logic knowledge base. Of course, this could be done by humans, but this approach would not only be very time-consuming, but also prone to errors. An automatic translation would thus be highly welcomed. On the other hand, a completely automatic translation would just mean that machines can consume knowledge in the way humans represent it, an assumption which is not reasonable.

However, one can still think about approaches which are *semi-automatic* in the sense that the results obtained from such a translation are preliminary, requiring further refinement, or that the translation procedure requires additional *assistance* by means of human experts.

An approach to achieve such a semi-automatic translation procedure, or *learning procedure*, has been made in [41]. There, the focus lies on extracting terminological knowledge of the form as shown in Equation (1.2) from some given *finite interpretation* $\mathcal{I}$. More precisely, the procedures described in [41] would automatically learn all GCIs which are *valid* in the data set $\mathcal{I}$. This approach makes use of the mathematical theory of *formal concept analysis*, a subfield of mathematical order theory, which has very close connections to description logics.

Of course, the GCIs learned this way may not be correct, in the sense that the GCI may hold in $\mathcal{I}$, but this data missed to contain some relevant counterexamples. In this case, we say that $\mathcal{I}$ is *incomplete*. One way to remedy this is to use an algorithm from formal concept analysis called *attribute exploration*, and generalize it to the setting of GCIs. Such a generalization, called *model exploration*, has been discussed in [41]. Within this algorithm, an expert (possibly human) is asked for the correctness of extracted GCIs, and if such a GCI is not correct, the expert has to provide *counterexamples* for it. This way, the problem of $\mathcal{I}$ being incomplete can be solved.

However, there are still problems with the original approach of [41], in particular concerning the *quality* of the data $\mathcal{I}$ from which GCIs are learned. The main problem here is that the data may contain *errors*. These errors may either cause otherwise valid GCIs not to be found, because these errors act as *false counterexamples*, or GCIs to be found which are not correct, because errors cause positive counterexamples to vanish. While the latter approach can in theory be handled by the attribute exploration approach sketched above, the former cannot, because the approach discussed in [41] will not even extract GCIs for which there may be false counterexamples in $\mathcal{I}$.

In this work, we want to extent the results obtained in [41] to this new setting of where the data $\mathcal{I}$ may contain errors. The main approach for this is to transfer the notion of *confidence* [1] from the area of data-mining to GCIs. Intuitively, this means that GCIs may have *few* errors in the data, as opposed to having none in the original approach of [41]. The notion of "few" is quantified by means of the confidence of the GCIs in the data.

In the following, we shall give a more in-depth discussion of what we want to do in this work. To this end, we first shall introduce description logics and formal concept analysis, in an exemplary and historic manner. Thereafter, we shall discuss the main results of [41] in more detail, and also briefly mention some other related work. Finally, we present the

main contributions of this work.

1.1. Description Logics

Description logics [12] are a family of logic-based knowledge representation formalisms, with a strong emphasis on well-defined semantics and practical reasoning procedures. The family of description logics contains various kinds of logical formalisms, varying in expressiveness and reasoning complexity, allowing users to choose the expressiveness they need, or the complexity they can afford in their respective applications.

The development of description logics [18] was motivated by earlier knowledge representation formalisms like *semantic networks* [91] or *frame* [71], whose semantics were highly ambiguous, and mostly depended on human interpretation or implementation details. The need for well-defined and predictable knowledge representation formalisms then led to the first logic-based systems [35], which however were incomplete [86].

The first description logics considered were relatively small fragments of first order logic, and already for them it could be shown that reasoning is intractable [34, 72]. One approach to remedy this was to investigate highly optimized reasoning algorithms, which behave well in practice. The most prominent class of such algorithms are *tableau algorithms*. These algorithms were first invented for the description logic $\mathcal{ALC}$ [60, 87] for the subsumption problem, and were thereafter extended to other, even more expressive logics. After a connection of $\mathcal{ALC}$ to multimodal logic $\mathsf{K}_{(\mathsf{m})}$ was discovered [85], it was seen that this tableau algorithm is actually a re-invention of the tableau algorithms used in modal logics. The development of description logics continued to investigate highly expressive description logics, whose expressiveness exceeds that of $\mathcal{ALC}$, but which still behave well in practice [62], and for which highly-optimized implementations exist [52, 90, 99]. This finally led to the adoption of the *Web Ontology Language OWL* by the W3C, which is based on the highly expressive description logic $\mathcal{SHOIN}$ [61].

The focus of description logics research departed from the sole focus on expressive description logics when, at the beginning of this millennium, it was discovered that for the inexpressive description logic $\mathcal{EL}$ reasoning is tractable [9, 36], and stays so when the expressiveness of $\mathcal{EL}$ is extended slightly [10, 11]. A practical relevance of these results is given by the fact that large biomedical ontologies can be reformulated as *description logic knowledge bases* (or *description logic ontologies*) using $\mathcal{EL}$ or such slight extensions of it. Examples for this are the *Systematized Nomenclature of Medicine–Clinical Terms*, the Gene Ontology [5], and large parts of the GALEN ontology [80].

The term "description" in "description logics" is motivated by the intention to use description logics to express knowledge about *concept descriptions*. For this, description logics provide a number of constructors, which can then be used to build concept descriptions from atomic *concept names* and binary *role names*. For example, the description logic $\mathcal{EL}$ provides the constructors *conjunction* ($\sqcap$) and *existential restriction* ($\exists$). Examples of $\mathcal{EL}$

concept descriptions are

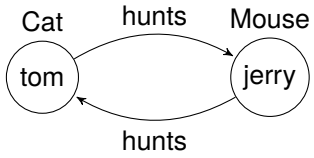
$$\mathsf{Cat},\ \mathsf{Cat} \sqcap \mathsf{Mouse},\ \exists \mathsf{hunts}.\mathsf{Mouse},$$

where Cat, Mouse are concept names, and hunts is a role name.

A description logic knowledge base formulated in $\mathcal{EL}$ consists, like most description logic knowledge bases, of two parts, namely an *ABox*, holding assertional knowledge, and a *TBox*, containing terminological knowledge. An example knowledge base is

$$\mathcal{K} = (\{\, \mathsf{Cat} \sqsubseteq \exists \mathsf{hunts}.\mathsf{Mouse} \,\}, \{\, \mathsf{Cat}(\mathsf{tom}) \,\}),$$

where the first entry denotes the TBox, and the second denotes the ABox. The semantics of knowledge bases is defined using *interpretations* $\mathcal{I}$, which can be thought of as directed edge- and vertex-labeled graphs. The labels of the vertices, which we shall call *elements* or *individuals*, are concept names, and the labels of the edges are role names. An interpretation is a model of a knowledge base if all elements *satisfy* the axioms contained in this knowledge base. For example, the interpretation



is a model of the knowledge base $\mathcal{K}$, since the element tom is labeled Cat, and every element which is labeled with Cat is connected to some element labeled Mouse via an edge labeled with hunts.

As soon as one has a description logic knowledge base, one can conduct *reasoning* with it, i. e. one can extract knowledge from the knowledge base that may be only contained in it implicitly. Two classical reasoning problems are *instance checking* and *subsumption*: given an *individual name a* and a concept description C, the instance checking problem is to ask whether a is an *instance* of C, i. e. whether a satisfies the concept description C in every interpretation. The subsumption problem is to ask, given two concept descriptions C and D, whether it is true that C is a *subconcept* of D, i. e. whether it is true in every interpretation that every element that satisfies C also satisfies D. Other reasoning problems are *knowledge base consistency* and *concept satisfiability*: a knowledge base is consistent if it has a model, and a concept description C is satisfiable with respect to a given knowledge base $\mathcal{K}$ if there exists a model of $\mathcal{K}$ containing elements that satisfy C. Deciding knowledge base consistency and concept satisfiability can help to ensure the correctness of the given knowledge base.

	small	medium	large	inner	outer	moon	nomoon
Mercury	×			×			×
Venus	×			×			×
Earth	×			×		×	
Mars	×			×		×	
Jupiter			×		×	×	
Saturn			×		×	×	
Uranus		×			×	×	
Neptune		×			×	×	
Pluto	×				×	×	

Figure 1.1.: Example Formal Context (taken from [103])

1.2. Formal Concept Analysis

Formal concept analysis [48] is a subfield of mathematical order theory, originally concerned with the study of properties of ordered structures called *complete lattices* by representing them in terms of so-called *formal contexts*. Since then, formal concept analysis has considerably broadened its scope, with connections to previously unrelated subjects such as logics [44, 77], data mining [105], machine learning [65], and artificial intelligence [41, 82]. Because of this, formal concept analysis today can be considered as a part of theoretical computer science, and thus it provides another link between computer science and mathematics.

The origin of formal concept analysis as it is used in this work can clearly be marked by the work of Wille [103], which introduced formal concept analysis as an approach to impose meaning on complete lattices by considering them as *hierarchies of concepts*. This work was motivated by previous results from Birkhoff [23], but also has a strong philosophical background [57, 102]. Another early work that included some of the ideas of formal concept analysis is [20].

The fundamental idea of formal concept analysis is to represent complete lattices by an *object-attribute-relationship*, which is expressed using *formal contexts*. These structures can be thought of as *tables of crosses*. An example of a formal context is depicted in Figure 1.1. This example formal context expresses an object-attribute-relationship between the *objects* being the known planets of the solar system (including Pluto), and the *attributes* being certain properties of these planets, like their size (small, medium, or large), their distance from the sun (being an inner or outer planet, i. e. having an orbit which is closer to the sun than the asteroid belt or not), and if they do or do not have a moon. A cross in this table then means that the object on the corresponding row *has* the attribute on the corresponding column. Thus, for example, Mercury is a small planet, and Pluto is an outer planet. The set

of all pairs of objects g and attributes m where g has the attribute m is called the *incidence relation* of the formal context.

Formally, a *formal context* $\mathbb{K}$ can be defined as a triple $\mathbb{K} = (G, M, I)$, where G and M are sets and $I \subseteq G \times M$. G is then called the set of *objects*, M is called the set of *attributes*, and the set I is called the *incidence*. An object $g \in G$ *has* an attribute $m \in M$ in $\mathbb{K}$ if and only if $(g, m) \in I$.

From such a formal context one can then extract *formal concepts*, which can be ordered in a natural way to yield the *concept lattice* of the formal context. In our example above, a formal concept which corresponds to the concept of a *medium-sized planet in our known solar system* would be the tuple

$$(\{\,\mathsf{Uranus}, \mathsf{Neptune}\,\}, \{\,\mathsf{medium}, \mathsf{moon}, \mathsf{outer}\,\}), \tag{1.3}$$

where the first set is called the *extent*, and the second set is called the *intent* of the formal concept.

Formal concepts can then be ordered by set-inclusion of their extents, and it can be shown that the thus-obtained ordered set is a complete lattice, the *concept lattice* of the formal context. The concept lattice that corresponds to our small example above is shown in Figure 1.2. This diagram also uses the usual, abridged annotation of concept lattices: a node v in the lattice diagram represents the formal concept whose extent consists of all objects which can be reached by an *descending path* in the diagram, starting from v. Likewise, the intent of v is the set of all attributes that can be reached by an *ascending path* in the diagram, starting from v. Thus, the gray-shaded node in Figure 1.2 is the formal concept of Equation (1.3).

One of the key results of formal concept analysis is that every complete lattice can be represented as a concept lattice of a suitably chosen formal context (this is the so-called *fundamental theorem of formal concept analysis*). A major direction of formal concept analysis research is to consider properties and operations of lattices, such as *distributivity*, *modularity*, *direct* and *semi-direct products*, and transfer them to corresponding properties and operations on the level of formal contexts. See [48] for more details on this.

Another very prominent research direction in formal concept analysis is the study of *implications* in formal contexts, which has been discussed as early as [103]. Observe that in our example formal context above, all outer planets have a moon. We can express this fact as saying that the implication

$$\{\,\mathsf{outer}\,\} \to \{\,\mathsf{moon}\,\}$$

holds in our formal context. Implications are similar to *functional dependencies* from the theory of databases [70], and also play a certain role in classical order theory [101].

One task is to compute the set of all valid implications of a given formal context. Since this set can be quite large, one usually wants to compute "small" sets of implications which are sufficient, called *bases*. One very prominent base is the so-called *canonical base* [51]

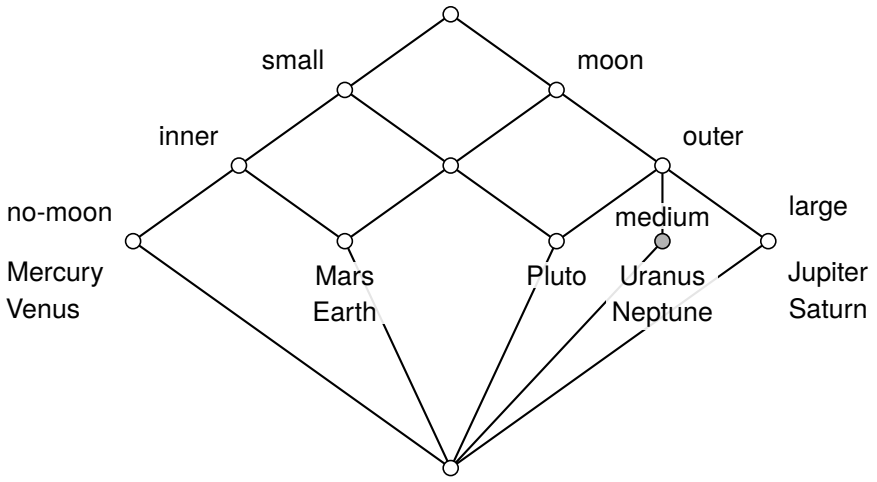


Figure 1.2.: Example Concept Lattice

(also *stem base*, *Duquenne-Guigues base*), which is a *minimal* base, i. e. a base with minimal cardinality. This base can be computed effectively [46, 75], however these algorithms are not efficient [40] with respect to the size of the input and the output. Certain complexity results suggest that efficiently computing the canonical base is not possible in general [19]. However, if this is really the case is an open research question. Therefore, other bases have been investigated, whose computation is algorithmically easier. An example is the base of *proper premises* [48], for which fast algorithms exist [84].

An algorithm that is related to the study of valid implications of formal contexts is *attribute exploration* [47, 48], which is an interactive process which extracts valid implications from *incomplete* data utilizing *expert interaction* to obtain missing facts. Within this process, an external expert is asked questions of the form

> Is the implication $A \to B$ valid?

The expert then can either accept this implication, or decline it by providing a *counterexample*. In this way, the expert enriches the currently known formal context by missing objects and their attributes. As soon as the process finishes, the set of confirmed implications represents the whole implicational knowledge represented by the expert. Moreover, it can be shown that the set of confirmed implications is the canonical base of the formal context which consists of the initially known objects together with all counterexamples provided by the expert. In this way, attribute exploration can be seen as a semi-automatic knowledge acquisition algorithm.

Attribute exploration has been a major focus of formal concept analysis research, and many extensions of this algorithm have been developed and discussed. Examples for this are

the inclusion of *background knowledge* [45, 92], *concept exploration* [93], *rule exploration* [106], *relational exploration* [82], exploration in the presence of *partial knowledge* [15, 37], and *model exploration* [41].

1.3. Extracting Terminological Knowledge from Relational Data

The main purpose of this work is to discuss a way to extract general concept inclusions from interpretations which are allowed to contain errors. The basis for our considerations are the results obtained by Baader and Distel [13, 14, 41] on computing *finite bases* of *valid* GCIs from finite interpretations. In the following, we shall briefly summarize the main results of this approach.

The goal of the work by Baader and Distel is to learn terminological knowledge about a certain *domain* of interest. For this we assume that we can represent this domain as a finite interpretation $\mathcal{I}$, i. e. our domain is representable as relational data. The terminological knowledge we are then interested in is the set $\mathrm{Th}(\mathcal{I})$ of valid GCIs of $\mathcal{I}$, using the description logic $\mathcal{EL}^\bot$.

A first problem here is that the set $\mathrm{Th}(\mathcal{I})$ is infinite in general: if $C \sqsubseteq D$ is valid in $\mathcal{I}$, and if r is a role name, then the GCI $\exists r.C \sqsubseteq \exists r.D$ is also valid in $\mathcal{I}$. To remedy this, Baader and Distel compute *finite bases* of $\mathrm{Th}(\mathcal{I})$, i. e. finite subsets $\mathcal{B}$ of $\mathrm{Th}(\mathcal{I})$ which are already *complete* for $\mathcal{I}$. In other words, finite bases $\mathcal{B}$ are finite sets of valid GCIs of $\mathcal{I}$, such that every GCI valid in $\mathcal{I}$ is already entailed by $\mathcal{B}$. One of the main results of their approach is that such finite bases always exist, and that they can be computed effectively.

To provide these results, Baader and Distel exploit the tight connection between the description logic $\mathcal{EL}^\bot$ and formal concept analysis, established by *model based most specific concept descriptions* and *induced formal contexts* of finite interpretations and sets of concept descriptions. More precisely, it can be shown that if $\mathbb{K}_\mathcal{I}$ denotes the induced formal context of $\mathcal{I}$, then every base of $\mathbb{K}_\mathcal{I}$ gives rise to a finite base of $\mathcal{I}$. A technical problem that arises here is that model-based most-specific concept descriptions are not necessarily expressible in the description logic $\mathcal{EL}^\bot$. Because of this, Baader and Distel consider the description logic $\mathcal{EL}^\bot_{\mathrm{gfp}}$, an extension of $\mathcal{EL}^\bot$ by *cyclic concept descriptions* using *greatest fixpoint semantics*.

The resulting algorithms for computing bases of finite interpretations are all effective. A preliminary implementation with applications to linked data has been presented in [33].

An additional issue addressed by Baader and Distel is the fact that the interpretation $\mathcal{I}$ may be *incomplete*, i. e. certain facts from the domain of interest may not be represented in it. The GCIs which are valid in $\mathcal{I}$ may not necessarily be valid in the domain of interest. This problem is very similar to the problem solved by attribute exploration, and indeed it can be shown that attribute exploration applied to the context $\mathbb{K}_\mathcal{I}$ can be transferred into an algorithm for *model exploration* of $\mathcal{I}$. This algorithm then allows to interactively compute bases of $\mathcal{I}$, allowing an expert to provide missing facts when required. In this way, learning GCIs which are invalid in the domain of interest can be avoided.

1.4. Other Related Work

The work of Baader and Distel is not the first attempt to bring together the worlds of description logics and formal concept analysis. Indeed, there have been several previous attempts to utilize formal concept analysis for description logic applications, and to add ideas from description logics to notions of formal concepts analysis.

One of the first results in description logics that utilizes formal concept analysis is the work of Baader [6]. In this work, Baader uses the attribute exploration algorithm on a special formal context to compute a minimal representation of the subsumption hierarchy between all conjunctions of the defined concept names of a given acyclic TBox $\mathcal{T}$ formulated in the logic $\mathcal{ALC}$. To this end, Baader extends the classical tableau algorithm for $\mathcal{ALC}$ to decide subsumption between *single* defined concept names [87], and extends it in such a way that it provides counterexamples to instances of the subsumption problem. These counterexamples are then collected into a suitable formal context, and from this context the attribute exploration algorithm eventually computes the canonical base. These implications give rise to a set of GCIs, which can then be used to decide subsumption between conjunctions of defined concept names of $\mathcal{T}$.

The results obtained by Baader have been further generalized by Stumme [95] to also include disjunction. For this, a generalization of attribute exploration called *distributive concept exploration* [94] has been used.

Another use of formal concept analysis for description logic applications is in *knowledge base completion* [15, 88], for which again attribute exploration was used. For this, the expert is asked GCIs of the form

$$\bigsqcap U \sqsubseteq \bigsqcap V,$$

where $U, V \subseteq M$ for some previously chosen set M of *interesting concepts*. The goal is then to ensure the given knowledge base is complete with respect to all these GCIs, i. e. the knowledge base should entail all such GCIs which are confirmed by the expert, and should contain counterexamples for all other GCIs of the above type. The greatest challenge for transferring attribute exploration to this setting is to deal with the *open world semantics* of description logic knowledge bases: if a fact is not entailed by the knowledge base, then this does not mean that the negated fact holds. For this, attribute exploration is generalized to work on *partial contexts*, i. e. formal contexts in which certain crosses are unknown. The resulting algorithm has been implemented as a plugin named *OntoComp* for the ontology editor *Protégé* [89].

A third prominent application of ideas of formal concept analysis in description logics is the work of Rudolph on *relational exploration* [81, 82]. In this approach the target description logic is $\mathcal{FLE}$, the extension of $\mathcal{EL}$ by value restriction ($\forall$). The domain of interest is not represented as an interpretation, but by means of *binary power context families* [79], which however can easily be considered as an interpretation. Then the exploration process is

conducted in several phases: in phase k, concept descriptions with *role depth* at most k are considered as attributes of the current formal context, and on this formal context attribute exploration is performed. Rudolph then shows that this process has to be considered only up to a certain maximal role depth, and that the resulting set of implications can be used to decide whether an arbitrary GCI $C \sqsubseteq D$ is valid or not in the domain represented by the expert. However, it is not shown whether and how this set of implications can be transferred into a base of the domain. Indeed, the decision procedure for checking whether $C \sqsubseteq D$ holds in the domain or not is rather complicated. Thus, no GCIs are learned from this approach, and thus it cannot be used to obtain terminological knowledge from relational data.

The aforementioned approach of *power context families* is an attempt to add description logic expressibility to the world of formal concept analysis. In its easiest form a formal context is equipped with a family of relations $\mathcal{R}$ on the object set to obtain a *relational context*. Based on this notion, *terminological attribute logic* [77] has been introduced that allows to define new attributes in terms of old ones, using the relational context to extend the incidence relation to the newly defined attributes. With this semantics, terminological attribute logic can be seen as a syntactic variant of $\mathcal{ALC}$ extended by inverse roles, negated roles and the identity role. Terminological attribute logic can also be defined for *many-valued contexts* to provide a method for *logical scaling* [78], which transforms many-valued contexts into the usual (two-valued) formal contexts in another way as the usual scaling approach of formal concept analysis.

Another approach to bring a flavor of description logics to formal concept analysis is *relational concept analysis* [53, 54]. The basic structure in this approach is a *relational context family*, which consists of a family of formal contexts and a set of relations between objects of possibly different contexts of this family. Using a method called *relational scaling* new attributes $r : C$ are added to the formal contexts, which roughly correspond to concept descriptions of the form $\forall r.C$ or $\exists r.C$, where however C is now a formal concept of a formal context of the family. In an iterative process, relational scaling is used to construct lattices from all formal contexts of the relational context family. These lattices can then be used to derive assertional and terminological knowledge formulated in the description logic $\mathcal{FLE}$.

Other research that is related to this work are approaches in formal concept analysis and description logics to tackle the problem of *uncertainty* and *vagueness*. In the area of formal concept analysis, the most notable and relevant work is the one by Luxenburger [68, 69], who considers implications in formal contexts together with an *accuracy* (confidence). Luxenburger then studies the problem of *realizations* of partial implications, which in terms of logic is just the question whether a set of partial implications is satisfiable. The results obtained here give a characterization in terms of *linear programs*. Furthermore, he studies the problem of finding *bases* of partial implications, and we shall use the main ideas in our later considerations. Luxenburger's ideas have been used in the area of data mining, for instance to obtain smaller representations of *association rules* [97].

Another prominent approach to handle knowledge that may not be completely correct is

to consider fuzzy extensions of the respective formalisms. Those exist for both description logics [27, 67] and formal concept analysis [21, 76], and are either based on Zadeh's original approach to fuzzy logics [104], or on the approach by Hájek [55], in which the semantics is defined using *t-norms* on lattices. In both cases, logical facts (implications, assertional knowledge, terminological knowledge) are annotated with *truth values* from an underlying lattice, and the semantics then allows to infer truth values for previously unknown facts, or at least bounds thereof.

Finally, a completely different approach to learning terminological knowledge has been developed recently [64], based on a general framework of *query learning* proposed by [2]. In contrast to the work by Baader and Distel, this approach tries to learn TBoxes $\mathcal{T}$ by posing queries to an *oracle*. These queries are either *entailment queries*, in which the oracle has to decide whether a proposed GCI is entailed by the TBox $\mathcal{T}$ to be learned, or *equivalence queries*, where the oracle has to decide whether a given TBox is equivalent to $\mathcal{T}$. The work [64] then considers the question for which description logic the TBox $\mathcal{T}$ can be learned in polynomial time. In particular, it is shown that if $\mathcal{T}$ is an *acyclic $\mathcal{EL}$-TBox*, then it cannot be learned in polynomial time.

Note that entailment queries are similar to the questions proposed to an expert during attribute exploration. In fact, it is possible to show that a certain special case of query learning [3] can be used to obtain an alternative computation of the canonical base [4].

1.5. Contributions

The main contributions of this thesis are the following.

1.5.1. Experiments with Extracting Valid GCIs

All results obtained by Baader and Distel are *effective*, meaning that the resulting algorithms can be implemented and applied to relational data. As a first contribution of this work, we shall present an implementation of these algorithms in Section 5.1. We then apply this implementation to relational data from the *DBpedia project* [25], which obtains its data by crawling *infoboxes* of Wikipedia articles. We shall see that the approach by Baader and Distel indeed is able to extract terminological knowledge from this data, and we shall discuss the corresponding outcome in detail.

The main observation of these experiments however is that the errors in the DBpedia data set inhibit GCIs from being found which otherwise would be relevant for the domain of the chosen data. The main example is that the GCI

$$\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}$$

stating the fact that every individual which has a child is a person, was not found during our experiments. The reason for this is that input data contained four counterexamples for

this GCI, which however were all erroneous. On the other hand, the input data contained 2547 *positive examples* for this GCI. This shows that the original approach of Baader and Distel is very sensitive to errors in the input data, even if they are comparably rare.

1.5.2. Extracting GCIs with High Confidence from Erroneous Data

A main result from our experiments is that errors in the input data can inhibit otherwise valid GCIs from being extracted. On the other hand, we can assume that the input data must be of sufficient *quality* to reasonably extract terminological knowledge from it, i. e. must not contain too many errors which are relevant for our computations. Based on this assumption we generalize the approach by Baader and Distel from computing bases of valid GCIs to computing bases of GCIs which are *almost valid* in the input data. To formalize the notion of being *almost valid*, we make use of the notion of *confidence* as it is used in data-mining, and transfer it to the setting of GCIs. Instead of computing bases of valid GCIs, we then want to compute bases of GCIs whose confidence is above a certain threshold c, for some chosen value $c \in [0, 1]$. Those GCIs we shall call GCIs *with high confidence*. We shall see that we can generalize most of the results of Baader and Distel accordingly, using Luxenburger's ideas on his investigation of partial implications in formal contexts. An example for such a result is the fact that bases of the induced context of a finite interpretation give rise to a base of the finite interpretation itself. We shall see that we can generalize this result to obtain that bases of implications with high confidence in the induced context give rise to bases of GCIs with high confidence in the finite interpretation. Finally, we shall apply our results to the data sets we used for our experiments, to show that our approach is able to handle certain errors in the input data.

1.5.3. Exploration by Confidence

The approach of considering GCIs with high confidence is a purely heuristic one: by considering GCIs with high confidence, we can ignore rare errors in the input data. However, our approach ignores *rare counterexamples* as well, i. e. counterexamples which are actually valid, but occur so infrequently that the confidence measures ignores them. In this case, an external source of information is needed that can distinguish between errors and rare counterexamples in the data. An example for such an external source of information could be a human expert, who then considers all such counterexamples manually, and decides whether they are valid or not.

Expert interaction can also be used to tackle another problem with data, namely its *incompleteness*: for certain GCIs relevant counterexamples may exist in the domain of interest, but may not be present in the given data set. In other words, we still assume that our domain is represented by an interpretation, the so-called *background interpretation*, but we only have access to some part of it. In this case, both the original approach by Baader and Distel, as well as our extension to GCIs with high confidence would extract those

GCIs for which relevant counterexamples are missing. We could use the external expert to avoid this issue, by querying this expert for possible counterexamples from the background interpretation for all GCIs we would extract.

This expert interaction may be expensive, however, and an algorithm that keeps this interaction to a minimum would be highly desirable. In the case of valid implications of a formal context, the attribute exploration algorithm would be such an algorithm, as it asks the expert a minimal number of different questions. Therefore, we want to generalize attribute exploration to the setting of GCIs with high confidence to obtain an algorithm that allows the use of an expert to distinguish between errors and rare counterexamples.

As a first step, we shall consider the easier problem of exploring *implications with high confidence* instead of GCIs. In this step, we shall generalize attribute exploration to *exploration by confidence*, where the algorithm not only asks the expert after implications which are valid in the current context, but also after those which only have high confidence. For this, we first investigate generalizations of attribute exploration which allow us to explore *sets of implications*. In these generalizations, a set $\mathcal{L}$ of implications can be specified for which the expert should decide which elements of $\mathcal{L}$ are valid. In attribute exploration, the set $\mathcal{L}$ is just the set of all valid implications of a given formal context. For the case of implications with high confidence, the set $\mathcal{L}$ is then just the set of all implications whose confidence is above a certain threshold c.

1.5.4. Model-Exploration by Confidence

The problem of incomplete data already arises in the case of valid GCIs, and to approach this problem Baader and Distel propose extensions of attribute exploration which also work with valid GCIs. One of these extensions, called *model exploration*, works mostly in the same way as attribute exploration does, with the difference that the expert now gets asked GCIs for confirmation instead of implications. Model exploration is essentially attribute exploration of the induced context of the given interpretation, although several technical problems have to be dealt with.

One of these problems it that the attribute set of the induced context depends on the background interpretation. Clearly, the background interpretation is not available during the exploration process. Model exploration solves this problem by computing the set of attributes of the induced context incrementally during the computation. Another problem is the way counterexamples have to be specified: because of the *closed world semantics* of interpretations, every counterexample provided by the expert has to be *complete*. This means that as soon as the expert wants to provide an element as a counterexample, all element which can be reached via directed edges from this elements in the background interpretation have to be provided as well. In other words, the counterexamples provided by the expert have to be *connected subinterpretations* of the background interpretation.

To generalize model exploration to GCIs with high confidence, we essentially follow the argumentation of model exploration. More precisely, we shall consider exploration

by confidence of the induced context of the given interpretation, and transform it to *model exploration by confidence*, in a very similar way as model exploration arises from attribute exploration of the induced context.

A notable difference to the argumentation of Baader and Distel is that we first have to generalize our results about computing bases of GCIs with high confidence to include the possibility that certain elements of the interpretation are *trusted*, in the sense that as soon as a trusted element is a counterexample for some GCI, this GCI is not considered any further, even if its confidence is high enough. The motivation for this generalization stems from the fact that we consider all counterexamples provided by the expert as valid.

CHAPTER 2

Formal Concept Analysis

In this section we shall introduce almost all notions from the area of formal concept analysis that are relevant for this work. One of the main aspects of formal concept analysis we are interested in are the methods it provides to extract *implicational dependencies* from data. To this end, we shall discuss in detail the notions of *implications* in formal contexts, *bases* of valid implications of formal contexts, and the computation of the *canonical base* as a particular example of a minimal base. This will be done in Sections 2.3 and 2.4. We shall also discuss related topics like *Galois connections* and *closure operators* (Section 2.2) and *attribute exploration* (Section 2.5). Before we can do so, however, we have to introduce some of the fundamental notions of formal concept analysis such as *formal contexts* and *contextual derivation*. This will be done in Section 2.1.

The introduction to formal concept analysis as given in this chapter is mainly based on [48]. Note that this introduction is specifically tailored towards the purpose of the whole work. As such, the following exposition is not complete in the sense that it discusses all the mathematical foundations of formal concept analysis. In particular, we may omit proofs or details of certain argumentations if they are not relevant for our work. In case more details are needed, we provide pointers to the literature where those details can be found.

2.1. Formal Contexts and Concept Lattices

We introduce the basic notions of formal concept analysis in this section. Most importantly, we shall discuss how formal concept analysis allows us to represent complete lattices in terms of *formal concept lattices* using the notion of *formal contexts*.

Let us briefly repeat the basic notions of order theory which are relevant for our further considerations. Let P be a set and let $\leqslant \subseteq P \times P$ be a binary relation on P. Then the pair $(P, \leqslant)$ is called a *(partially) ordered set* if $\leqslant$ is reflexive, transitive and antisymmetric. Such structures can be visualized in terms of *order diagrams* (often called *Hasse diagrams*) if they are finite (and not too large). For this we call two elements $x, y \in P$ with $x < y$ *directly neighbored* in $(P, \leqslant)$ if and only if there does not exist an element $z \in P$ such that $x < z < y$.

Then to visualize $(P, \leqslant)$ we mark for every element $x \in P$ a node v_x on the plane such that whenever $x < y$ it is true that the ordinate (the second coordinate) of v_x is strictly smaller then the one of v_y. Then, we draw for every two elements x, y with $x < y$ which are directly neighbored in $(P, \leqslant)$ an undirected line from v_x to v_y.

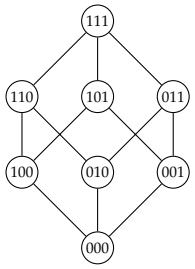
Observe that this construction is not unique, and there are many different possibilities (good and bad) to visualize ordered sets in this way. Also note that the naming v_x of vertices for elements x is arbitrary and can be chosen as it suits.

2.1.1 Example Let us consider the set $\{1, 2, 3\}$ with the usual order $\leqslant$ on natural numbers. Then a line diagram of this ordered set is

We can readily read of this diagram that $1 < 2$ and $2 < 3$, because there are lines connecting the corresponding vertices. But we can also see that $1 < 3$ because there is an *ascending path* from 1 to 3. ♢

More generally, in a line diagram of an ordered set $(P, \leqslant)$, two elements $x, y \in P$ satisfy $x \leqslant y$ if and only if there is an ascending path from x to y in the line diagram (where also paths of length 0 are allowed).

2.1.2 Example Let $P = \{1, 2, 3\}$ and consider the ordered set $(\mathfrak{P}(\{1, 2, 3\}), \subseteq)$, where $\mathfrak{P}(\{1, 2, 3\})$ denotes the set of all subsets of $\{1, 2, 3\}$. This set can be visualized as a line diagram as follows:



Here we denote subsets of $\{1, 2, 3\}$ by sequences of 0 and 1, meaning that the first position is 1 if and only if the number 1 is an element of the corresponding subset, and so on. Then 101 corresponds to the set $\{1, 3\}$. ♢

An element $x \in P$ is said to be the *smallest element* of $(P, \leqslant)$ if and only if $x \leqslant y$ is true for all $y \in P$. Likewise, x is the *greatest element* of $(P, \leqslant)$ if and only if $y \leqslant x$ is true for all

$y \in P$. Note that neither smallest nor greatest elements have to exist in P. However, if these elements exist, they are unique.

Let $Q \subseteq P$. An element $x \in Q$ is said to be the *smallest element* in Q (with respect to $(P, \leqslant)$) if and only if $x \leqslant y$ is true for all $y \in Q$; *greatest elements* in Q are defined likewise. Again, neither smallest nor greatest elements in Q have to exist.

Let $x \in P$. The *order-ideal* $\downarrow x$ and the *order-filter* $\uparrow x$ of x in $(P, \leqslant)$ are defined as

$$\downarrow x := \{\, y \in P \mid y \leqslant x \,\},$$
$$\uparrow x := \{\, y \in P \mid x \leqslant y \,\}.$$

In other words, $\downarrow x$ contains all elements which are *below* x in $(P, \leqslant)$, and $\uparrow x$ contains all elements which are *above* x in $(P, \leqslant)$.

Let again $Q \subseteq P$. Then the sets Q^* and Q_* defined as

$$Q^* := \bigcap_{q \in Q} \uparrow q,$$
$$Q_* := \bigcap_{q \in Q} \downarrow q$$

are called the set of *upper bounds* and *lower bounds* of Q in $(P, \leqslant)$, respectively, where we employ the convention that $\bigcap \varnothing = P$. If Q^* has a smallest element (a *least upper bound*) in $(P, \leqslant)$, it is called the *supremum* of Q in $(P, \leqslant)$ and is denoted by $\sup Q$. Likewise, if Q_* has a greatest element (a *greatest lower bound*) in $(P, \leqslant)$, then it is called the *infimum* of Q in $(P, \leqslant)$ and is denoted by $\inf Q$.

Note again that neither infimum nor supremum have to exist in $(P, \leqslant)$.

2.1.3 Example Let $P = \{\, a, b, c \,\}$ and let $\leqslant$ be given by the smallest order relation that satisfies $a < b$ and $a < c$. Then $\inf\{\, b, c \,\}$ exists in $(P, \leqslant)$ and is equal to a. However, $\sup\{\, b, c \,\}$ does not exist in $(P, \leqslant)$, as $\uparrow b \cap \uparrow c = \varnothing$. ♢

Structures in which supremum and infimum always exist for *finite* sets Q are called *lattices*. If the sets Q can be chosen arbitrary, then we call such a structure a *complete lattice*.

2.1.4 Definition (Lattice) Let $\underline{L} = (L, \leqslant)$ be an ordered set. Then $\underline{L}$ is called a *lattice* if and only if for each non-empty finite $Q \subseteq L$ there exist both $\sup Q$ and $\inf Q$ in $\underline{L}$. If for all $Q \subseteq L$ there exist both $\sup Q$ and $\inf Q$ in $\underline{L}$, then $\underline{L}$ is called a *complete lattice*. ♢

From time to time we may also use another notation for inf and sup: if Q is finite, then $Q = \{\, q_1, \ldots, q_n \,\}$ and we may write

$$q_1 \wedge \cdots \wedge q_n \quad \text{instead of} \quad \inf Q,$$
$$q_1 \vee \cdots \vee q_n \quad \text{instead of} \quad \sup Q.$$

It is also common to write $\bigwedge Q$ instead of $\inf Q$, and $\bigvee Q$ instead of $\sup Q$.

Note that every finite lattice is also a complete lattice. Moreover, every complete lattice has a smallest and greatest element, given by $\sup \varnothing$ and $\inf \varnothing$, respectively. Furthermore, every ordered set $(P, \leqslant)$ in which the supremum exists for each $Q \subseteq P$ is already a complete lattice, as the infimum is then given by

$$\inf Q = \sup \{\, x \in P \mid \forall y \in Q \colon x \leqslant y \,\}.$$

The same is of course true if supremum and infimum are exchanged.

The ordered sets from Examples 2.1.1 and 2.1.2 are lattices, but not the one from 2.1.3. More generally, if P is a set, then $(\mathfrak{P}(P), \subseteq)$ is always a complete lattice.

The study of lattices as mathematical structures has received much interest over the last decades, and thus constitutes a major branch of order theory [23, 38, 49]. Additionally, (complete) lattices also allow for a quite natural interpretation as a hierarchy of *generalizations* and *specializations*: an element x is below an element y if and only if x is more *special* than y, or alternatively, if y is more *general* than x. Then for a set Q of elements its supremum $\sup Q$ can be thought of as a *most-specific generalization* of all elements in Q, and $\inf Q$ can be seen likewise as the *most general specialization* of all elements in Q.

Formal concept analysis now provides an approach to understand complete lattice in terms of this interpretation, by representing these lattices in terms of *objects* and their *attributes*. For this, we need to introduce the notion of a *formal context*.

2.1.5 Definition (Formal Context) A *formal context* $\mathbb{K}$ is constituted of three sets G, M, I, where $I \subseteq G \times M$. Formally, a formal contexts $\mathbb{K}$ is a triple $\mathbb{K} = (G, M, I)$ where G, M are sets and $I \subseteq G \times M$. We shall call G the set of *objects* of $\mathbb{K}$, M the set of *attributes* of $\mathbb{K}$ and I the *incidence* of $\mathbb{K}$. Two formal contexts are equal if and only if their sets of objects, attributes and their incidences are equal, respectively. ♢

Formal contexts can be thought of as simple data structures which record, for a given set of objects G and a given set of attributes M, for each object $g \in G$ the set of attributes from M that g has. More precisely, we shall say that in a formal context $\mathbb{K}$ an object $g \in G$ *has* an attribute m if and only if $(g, m) \in I$.

2.1.6 Example Let us consider a small toy example $\mathbb{K}_{\mathsf{TNG}}$ to illustrate the definition of a formal context. As sets of objects we choose some fictional characters from *Star Trek: The Next Generation*, namely

$$G := \{\, \mathsf{Picard}, \mathsf{Worf}, \mathsf{Data}, \mathsf{BorgQueen} \,\}.$$

As sets of attributes we choose

$$M := \{\, \mathsf{Human}, \mathsf{Honorable}, \mathsf{Artificial}, \mathsf{StarFleet} \,\}.$$

To illustrate the incidence relation of our example formal context we make use of a *cross table*, i. e. we depict $\mathbb{K}_{\mathsf{TNG}}$ as a table where the rows are labeled with objects and the columns are labeled with attributes. Then in every cell we write a cross if and only if the object labeling the corresponding row has the attribute labeling the corresponding column.

$\mathbb{K}_{\mathsf{TNG}}$	Human	Honorable	Artificial	StarFleet
Picard	×	×		×
Worf		×		×
Data		×	×	×
BorgQueen			×	

Then, for example, BorgQueen has the attribute Artificial, but not Honorable. ♢

To now expose the connection between formal contexts on the one hand and complete lattices on the other we shall introduce the *derivation operators* in formal contexts.

2.1.7 Definition (Contextual Derivation) Let $\mathbb{K} = (G, M, I)$ be a formal context and let $A \subseteq G$ be a set of objects. Then the set of *common attributes* A' of A is defined to be

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

Likewise, for a set $B \subseteq M$ of attributes we define the set B' of *shared objects* as

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

The functions $A \mapsto A'$ and $B \mapsto B'$ are called the *derivation operators* of $\mathbb{K}$, and the sets A' and B' are called the *derivations* of A and B in $\mathbb{K}$, respectively. ♢

For $(A')'$ we may also simply write A''.

Note that both derivation operators are denoted by $(\cdot)'$, which usually does not lead to confusion, as it is most often clear from the context whether we deal with a set of objects or a set of attributes from which we want to compute its derivation. If it nevertheless happens that a single name for the derivation operator leads to confusion, then we shall locally introduce separate names for both of them.

What occurs more often, for example in Chapters 6 and 7, is the derivation of sets in *different formal contexts*. For example we may have given two formal contexts $\mathbb{K}_1 = (G_1, M_1, I_1)$ and $\mathbb{K}_2 = (G_2, M_2, I_2)$ and a set $A \subseteq M_1 \cap M_2$. When writing A', it is not clear in which context we do the derivation. To remedy this, we shall add a subscript to the set A to make clear of which context we consider it as a set of attributes: $A_{\mathbb{K}_1}$ denotes the set A considered as a set of attributes in $\mathbb{K}_1$, and likewise $A_{\mathbb{K}_2}$. While this notation is not useful as it stands, it becomes handy if we consider derivations of A: $(A_{\mathbb{K}_1})'$ denotes the derivation of A in the formal context $\mathbb{K}_1$, while $(A_{\mathbb{K}_2})'$ does the same for the formal context $\mathbb{K}_2$. Of course, we can drop the parentheses if that does not lead to ambiguity, and write $A'_{\mathbb{K}_1}$ and $A'_{\mathbb{K}_2}$ instead. Of course, the same can be done for sets of objects. In particular, instead of writing $(A'_{\mathbb{K}_1})'_{\mathbb{K}_1}$ we shall often only write $A''_{\mathbb{K}_1}$.

2.1.8 Example Consider our Star Trek example context from 2.1.6 and let $A := \{ \mathsf{Human} \}$. Then $A' = \{ \mathsf{Picard} \}$, $A'' = \{ \mathsf{Human}, \mathsf{Honorable}, \mathsf{StarFleet} \}$, and $A''' = \{ \mathsf{Picard} \} = A'$.$\diamondsuit$

The case that $A''' = A'$ is true in the previous example is not a coincidence, but an instance of a more general result: the derivation operators of a formal context form a *Galois connection* between the ordered sets $(\mathfrak{P}(G), \subseteq)$ and $(\mathfrak{P}(M), \subseteq)$.

2.1.9 Lemma *Let $\mathbb{K} = (G, M, I)$ be a formal context and let $A \subseteq M$ and $B \subseteq G$. Then it is true that*

$$A \subseteq B' \iff B \subseteq A'. \tag{2.1}$$

From this the following properties of the derivation operators of $\mathbb{K}$ can be derived: let $A, A_1, A_2 \subseteq M$ and $B, B_1, B_2 \subseteq G$. Then

i. $A_1 \subseteq A_2 \implies A_2' \subseteq A_1'$,

ii. $B_1 \subseteq B_2 \implies B_2' \subseteq B_1'$,

iii. $A \subseteq A''$,

iv. $B \subseteq B''$,

v. $A' = A'''$,

vi. $B' = B'''$.

In the proof of the lemma we shall only show Equation (2.1), as the remaining claims then are an immediate consequence of the more general result Lemma 2.2.3 on Galois connections.

Proof (Lemma 2.1.9) We can easily compute that

$$\begin{aligned} A \subseteq B' &\iff \forall m \in A \colon m \in B' \\ &\iff \forall m \in A \; \forall g \in B \colon (g, m) \in I \\ &\iff \forall g \in B \; \forall m \in A \colon (g, m) \in I \\ &\iff \forall g \in B \colon g \in A' \\ &\iff B \subseteq A' \end{aligned}$$

which shows the claim. □

Another useful property of the derivation operators is

$$A' = \bigcap_{a \in A} \{ a \}'$$

for $A \subseteq M$. This can easily be generalized into the following statement.

2.1.10 Lemma *Let $\mathbb{K} = (G, M, I)$ be a formal context, let $A \subseteq M$ and let $(B_j \mid j \in J)$ be a family of sets $B_j \subseteq M$ such that*

$$A = \bigcup_{j \in J} B_j.$$

Then

$$A' = \bigcap \{ B_j' \mid j \in J \}. \tag{2.2}$$

In particular, for every $\mathcal{A} \subseteq \mathfrak{P}(M)$ it is true that

$$\bigcap_{A \in \mathcal{A}} A' = (\bigcup_{A \in \mathcal{A}} A)'. \tag{2.3}$$

Of course, the same is true if A and all B_j are sets of objects instead of sets of attributes.

Proof (Lemma 2.1.10) Since $B_j \subseteq A$ for all $j \in J$, we can infer from Lemma 2.1.9 that $A' \subseteq B_j'$. It therefore suffices to show that $A' \supseteq \bigcap \{ B_j' \mid j \in J \}$.

To this end, let $g \in \bigcap \{ B_j' \mid j \in J \}$. Then $g \in B_j'$ for each $j \in J$, and therefore $\{ g \}' \supseteq B_j$, again for all $j \in J$. Since $A = \bigcup_{j \in J} B_j$, we obtain from this that $\{ g \}' \supseteq A$, and thus $g \in A'$, as required. □

We have claimed earlier that there exists a close connection between complete lattices on the one hand and formal contexts on the other. Having introduced the derivation operators, we are now able to expose this connection. To this end, we shall introduce the notion of *formal concepts* of a formal context.

2.1.11 Definition (Formal Concept) Let $\mathbb{K} = (G, M, I)$ be a formal context. Then a *formal concept* of $\mathbb{K}$ is a pair (A, B) such that $A \subseteq G, B \subseteq M$, and $A' = B, B' = A$ holds. The first entry of a formal concept is called its *extent*, and the second one called its *intent*. The set of all formal concepts of $\mathbb{K}$ is denoted by $\mathfrak{B}(\mathbb{K})$. ♢

Note that for each $B \subseteq M$, the pair (B', B'') is a formal concept of $\mathbb{K}$. Moreover, a set $B \subseteq M$ is an intent of $\mathbb{K}$ if and only if $B = B''$: if $B = B''$, then (B', B) is a formal concept of $\mathbb{K}$, and if (A, B) is a formal concept of $\mathbb{K}$, then $B'' = A' = B$. The same is of course true for $A \subseteq G$: A is an extent of $\mathbb{K}$ if and only if $A = A''$.

Formal concepts have a strong philosophical motivation, as they are an attempt to formalize the rather vague notion of a *concept*. This formalization is based on the perception that every formal concept is uniquely determined by the objects which are instances of it, its *extension*, as well as by a characterization in terms of attributes, its *intension*. We shall, however, not pursue this philosophical motivation any further here, as it is not immediately relevant for our work. See [48] for further discussion and references.

2.1.12 Example Two formal concepts of $\mathbb{K}_{\mathsf{TNG}}$ from Example 2.1.6 are

$$(\{\,\mathsf{Picard}\,\}, \{\,\mathsf{Human}, \mathsf{Honorable}, \mathsf{StarFleet}\,\}),$$
$$(\{\,\mathsf{Picard}, \mathsf{Worf}, \mathsf{Data}\,\}, \{\,\mathsf{Honorable}, \mathsf{StarFleet}\,\}).$$

The first formal concept could be said to represent the concept of an *honorable human*, which in $\mathbb{K}_{\mathsf{TNG}}$ has Picard as its only instance. The second formal concept describes everything *honorable*, having as extension Picard, Worf and Data. ♢

Formal concepts can be ordered by *generality*: a formal concept (A_1, B_1) is *more general* than another formal concept (A_2, B_2) if it covers more objects, i. e. if

$$A_1 \subseteq A_2.$$

2.1.13 Definition (Concept Lattice) Let $\mathbb{K} = (G, M, I)$ be a formal context. We define the relation $\leqslant$ on $\mathfrak{B}(\mathbb{K})$ by

$$(A_1, B_1) \leqslant (A_2, B_2) \;:\Longleftrightarrow\; A_1 \subseteq A_2.$$

The structure $\underline{\mathfrak{B}}(\mathbb{K}) := (\mathfrak{B}(\mathbb{K}), \leqslant)$ is called the *concept lattice* of $\mathbb{K}$. ♢

By Lemma 2.1.9 we can observe that

$$(A_1, B_1) \leqslant (A_2, B_2) \iff B_2 \subseteq B_1,$$

as $B_1 = A_1'$ and $B_2 = A_2'$. Furthermore, it is rather easy to see that the relation $\leqslant$ from Definition 2.1.13 is an order relation on $\mathfrak{B}(\mathbb{K})$. However, even more is true, namely that $\underline{\mathfrak{B}}(\mathbb{K})$ is indeed a complete lattice.

2.1.14 Theorem *Let $\mathbb{K}$ be a formal context. Then $\underline{\mathfrak{B}}(\mathbb{K})$ is a complete lattice, and for formal concepts $((A_j, B_j) \mid j \in J)$ of $\mathbb{K}$ we have*

$$\sup\{\,(A_j, B_j) \mid j \in J\,\} = \Big(\big(\bigcap_{j\in J} B_j\big)', \bigcap_{j\in J} B_j\Big),$$
$$\inf\{\,(A_j, B_j) \mid j \in J\,\} = \Big(\bigcap_{j\in J} A_j, \big(\bigcap_{j\in J} A_j\big)'\Big).$$

On the other hand, every complete lattice can be represented as a concept lattice of a suitably chosen formal context. To formalize this correctly, we shall introduce the notion of an *order isomorphism* between two ordered sets.

2.1.15 Definition (Order Isomorphism) Let $\underline{P} = (P, \leqslant_1)$ and $\underline{Q} = (Q, \leqslant_2)$ be two ordered sets. A bijective mapping $\varphi\colon P \to Q$ is called an *order isomorphism* if and only if φ is *order-preserving* and *order-reflecting*, i. e. it is true for all $a, b \in P$ that

$$a \leqslant_1 b \iff \varphi(a) \leqslant_2 \varphi(b).$$ ♢

The statement now is that every complete lattice is *order-isomorphic* to some concept lattice.

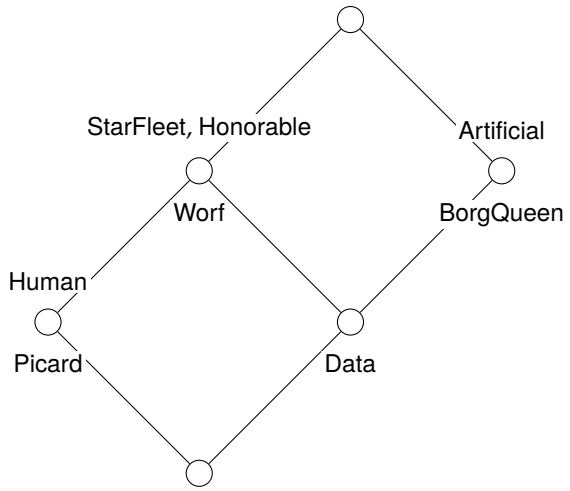


Figure 2.1.: Drawing of the concept lattice of $\mathbb{K}_{\mathsf{TNG}}$

2.1.16 Theorem *Let $\underline{V} = (V, \leqslant_V)$ be a complete lattice. Then the mapping $\varphi\colon V \to \mathfrak{B}(\mathbb{K})$ defined by*

$$\varphi(v) := (\{\, w \in V \mid w \leqslant_V v \,\}, \{\, w \in V \mid v \leqslant_V w \,\})$$

is an order isomorphism between $\underline{V}$ and $\underline{\mathfrak{B}}(V, V, \leqslant_V)$.

Both Theorem 2.1.14 and Theorem 2.1.16 are actually part of the Basic Theorem of formal concept analysis [48, Theorem 3]. We shall not give a proof of it here, as it is not within the scope of this work.

2.1.17 Example Let us consider the formal context $\mathbb{K}_{\mathsf{TNG}}$ of Example 2.1.6 again, and let us draw the concept lattice of $\mathbb{K}_{\mathsf{TNG}}$ in form of a line diagram. There are six formal concepts of $\mathbb{K}_{\mathsf{TNG}}$, and they can be depicted as shown in Figure 2.1.

The line diagram of Figure 2.1 uses an abridged annotation which is common for concept lattices: instead of annotating every node in the diagram with the formal concept it represents (which would yield an unreadable diagram), we only write every object g of $\mathbb{K}_{\mathsf{TNG}}$ below the *smallest* formal concept that has g in its extent. This formal concept always exists by Theorem 2.1.14. Then, by the definition of the order on formal concepts, every formal concept which can be reached from the one labeled with g by an *ascending* path in the line diagram has g in its extent, and all other formal concepts do not. For example, the formal concept labeled with Picard has this object in its extent, as well as the two formal concepts above it. No other formal concept has Picard in its extent.

Likewise, we write every attribute m of $\mathbb{K}_{\mathsf{TNG}}$ only at the *largest* formal concept that has this attribute in its intent. Then every formal concept that can be reached from the

one labeled with m by a *descending* path in the line diagram has m in its intent, and all the others do not. Therefore, the node labeled with Artificial has this attribute in its intent, as has the formal concept labeled with Data and the bottom concept. No other formal concept has Artificial in its intent. ♢

2.2. Galois Connections and Closure Operators

For the proof of Lemma 2.1.9 we invoked some general arguments from the theory of Galois connections between ordered sets. The notion of a Galois connection is fundamental for order theory, and is closely connected to other important concepts such as closure operators. As both Galois connections and closure operators play an important role in this work, we shall review their general theory in this section, to the extent needed in this work.

2.2.1 Definition (Galois Connection) Let $\underline{P} = (P, \leqslant_P), \underline{Q} = (Q, \leqslant_Q)$ be two ordered sets, and let $\varphi\colon P \to Q$ and $\psi\colon Q \to P$ be two mappings. Then the tuple $(\underline{P}, \underline{Q}, \varphi, \psi)$ is called a *Galois connection* between $\underline{P}$ and $\underline{Q}$ if and only if

$$x \leqslant_P \psi(y) \iff y \leqslant_Q \varphi(x) \tag{2.4}$$

is true for all $x \in P$, $y \in Q$. ♢

Note that this form of a Galois connection is sometimes called an *antitone* Galois connection, as the position of the elements x and y is reversed. There is the corresponding notion of an *isotone* Galois connection, where Equation (2.4) is replaced by

$$x \leqslant_P \psi(y) \iff \varphi(x) \leqslant_Q y. \tag{2.5}$$

Of course, both notions are closely related: if we denote with $\underline{Q}^d = (Q, \leqslant_Q^{-1})$ the *dual* of the ordered set $\underline{Q}$, then $(\underline{P}, \underline{Q}, \varphi, \psi)$ is an antitone Galois connection if and only if $(\underline{P}, \underline{Q}^d, \varphi, \psi)$ is an isotone Galois connection.

2.2.2 Example Two examples of Galois connections are the following.

i. If φ is an order isomorphism from $\underline{P} = (P, \leqslant_P)$ to $\underline{Q} = (Q, \leqslant_Q)$, then $(\underline{P}, \underline{Q}, \varphi, \varphi^{-1})$ is an isotone Galois connection, because

$$\begin{aligned} x \leqslant_P \varphi^{-1}(y) &\iff \varphi(x) \leqslant_Q \varphi(\varphi^{-1}(y)) \\ &\iff \varphi(x) \leqslant_Q y \end{aligned}$$

is true for all $x \in P$ and $y \in Q$.

On the other hand, if $\varphi\colon P \to Q$ is a bijective mapping such that $(\underline{P}, \underline{Q}, \varphi, \varphi^{-1})$ is an isotone Galois connection, then clearly φ is an order isomorphism by Lemma 2.2.3. In this sense, Galois connections are a generalization of order isomorphisms.

ii. If $\mathbb{K} = (G, M, I)$ is a formal context, then the derivation operators $(\cdot)' \colon \mathfrak{P}(G) \to \mathfrak{P}(M)$ and $(\cdot)' \colon \mathfrak{P}(M) \to \mathfrak{P}(G)$ form an antitone Galois connection by Lemma 2.1.9. ♢

Indeed, the converse of the last example is also true to some extent, i. e. every antitone Galois connection between powerset lattices $(\mathfrak{P}(G), \subseteq), (\mathfrak{P}(M), \subseteq)$ can be represented by a formal context $\mathbb{K}$, such that the derivation operators of $\mathbb{K}$ are just the mappings from the Galois connections. However, we shall not go into details here, as this is not relevant for the purpose of our work. See [48] for more details.

Instead, we shall review some useful properties of antitone Galois connections.

2.2.3 Lemma *Let $(\underline{P}, \underline{Q}, \varphi, \psi)$ be an antitone Galois connection between the ordered sets $\underline{P} = (P, \leqslant_P)$ and $\underline{Q} = (Q, \leqslant_Q)$. Then the following statements hold for all $a_1, a_2 \in P$ and $b_1, b_2 \in Q$:*

i. $a_1 \leqslant_P a_2 \implies \varphi(a_2) \leqslant_Q \varphi(a_1)$,

ii. $b_1 \leqslant_Q b_2 \implies \psi(b_2) \leqslant_P \psi(b_1)$,

iii. $a_1 \leqslant_P \psi(\varphi(a_1))$,

iv. $b_1 \leqslant_Q \varphi(\psi(b_1))$,

v. $\varphi(a_1) = \varphi(\psi(\varphi(a_1)))$,

vi. $\psi(b_1) = \psi(\varphi(\psi(b_1)))$.

Proof We only show statements (i), (iii) and (v), as the others follow from similar arguments.

We immediately obtain the truth of (iii), since from $\varphi(a_1) \leqslant_Q \varphi(a_1)$ we can infer $a_1 \leqslant_P \psi(\varphi(a_1))$ by Equation (2.4) of a Galois connection.

Then for (i) we assume $a_1 \leqslant_P a_2$ and obtain from (iii) that then $a_1 \leqslant_P \psi(\varphi(a_2))$. Using again the definition of a Galois connection we obtain that $\varphi(a_2) \leqslant_Q \varphi(a_1)$.

Finally, for (v) we already know that $\varphi(\psi(\varphi(a_1))) \leqslant_Q \varphi(a_1)$ is true. On the other hand, $\psi(\varphi(a_1)) \leqslant_P \psi(\varphi(a_1))$, so by the definition of a Galois connection, we obtain $\varphi(a_1) \leqslant_Q \varphi(\psi(\varphi(a_1)))$. Since $\leqslant_Q$ is antisymmetric, equality follows. □

Galois connections are closely related to the notion of *closure operators*.

2.2.4 Definition (Closure Operator) Let $\underline{P} = (P, \leqslant_P)$ be an ordered set, and let $c \colon P \to P$ be a mapping. Then c is called a *closure operator* on $\underline{P}$ if and only if

i. $a \leqslant_P c(a)$ for all $a \in P$ (c is *extensive*),

ii. $a \leqslant_P b \implies c(a) \leqslant_P c(b)$ for all $a, b \in P$ (c is *monotone*), and

iii. $c(a) = c(c(a))$ for all $a \in P$ (c is *idempotent*).

The element $c(a)$ for $a \in P$ is called the *closure* of a under c. An element $a \in P$ is called *closed* under c if and only if $a = c(a)$. ♢

Closure operators arise naturally in many situations, and we shall encounter them in the next section when we introduce implications. Moreover, closure operators and Galois connections always appear together: If $(\underline{P}, \underline{Q}, \varphi, \psi)$ is a Galois connection, then the mapping $\psi \circ \varphi$ is a closure operator on $\underline{P}$, and the mapping $\varphi \circ \psi$ is a closure operator on $\underline{Q}$. This is true because we know that $\psi \circ \varphi$ is extensive (2.2.3, iii), monotone (2.2.3, i and ii) and idempotent (2.2.3, v), i. e. a closure operator. Showing that $\varphi \circ \psi$ is a closure operator as well can be done similarly.

Conversely, if c is a closure operator on the ordered set $\underline{P}$, then there always exists a Galois connection $(\underline{P}, \underline{Q}, \varphi, \psi)$ such that $c = \psi \circ \varphi$. See [48] for more details on this.

There are two interesting properties of closure operators which are also relevant for our purpose. The first observation is that if c is a closure operator on the ordered set $\underline{P}$ then the infimum $c(x) \wedge c(y)$ is again closed under c for all $x, y \in P$. This is because on the one hand we have

$$c(c(x) \wedge c(y)) \leqslant_P c(c(x)) = c(x),$$
$$c(c(x) \wedge c(y)) \leqslant_P c(c(y)) = c(y),$$

by monotonicity and idempotency of c. Therefore $c(c(x) \wedge c(y)) \leqslant_P c(x) \wedge c(y)$. On the other hand, $c(x) \wedge c(y) \leqslant_P c(c(x) \wedge c(y))$ because c is extensive, and thus

$$c(c(x) \wedge c(y)) = c(x) \wedge c(y), \tag{2.6}$$

i. e. $c(x) \wedge c(y)$ is closed. It is also not hard to see that this argumentation can be lifted to arbitrary infima, i. e. for all $Q \subseteq P$, the element

$$\bigwedge_{x \in Q} c(x)$$

is closed under c. In other words, the closed sets of c always form a complete sublattice of $\underline{P}$. In particular, for all $x \in P$ there always exists a smallest element $z \in P$ above x which is closed under c, namely

$$z = \bigwedge_{y \in P, x \leqslant_P c(y)} c(y) = c(x).$$

2.3. Implications

Let us recall Example 2.1.6, were we had considered the formal context $\mathbb{K}_{\mathsf{TNG}}$ with

$\mathbb{K}_{\mathsf{TNG}}$	Human	Honorable	Artificial	StarFleet
Picard	×	×		×
Worf		×		×
Data		×	×	×
BorgQueen			×	

In this formal context we see that whenever an object has the attribute StarFleet, it also has the attribute Honorable. This expresses a certain dependency between these two attributes, in the sense that StarFleet *implies* Honorable in the formal context $\mathbb{K}_{\mathsf{TNG}}$. Knowing such *implicational dependencies* between attributes in a formal context can be very helpful, for example for reducing databases by transferring them into suitable normal forms or, as in our case, for learning knowledge from data.

We will model such implicational dependencies by means of *implications* on sets, which will be *valid* in a formal context.

2.3.1 Definition (Implications, Validity in Formal Contexts) Let M be a set. Then an implication $A \to B$ on M is constituted of two sets $A, B \subseteq M$, where A is called the *premise* of $A \to B$, and B is called the *conclusion* of $A \to B$. The set of all implications on M is denoted by $\mathrm{Imp}(M)$.

Let $\mathbb{K} = (G, M, I)$ be a formal context. An implication $(A \to B) \in \mathrm{Imp}(M)$ is said to be *valid* in $\mathbb{K}$ (or: *holds* in $\mathbb{K}$) if and only if $A' \subseteq B'$. We shall write $\mathbb{K} \models (A \to B)$ in this case, and $\mathbb{K} \models \mathcal{L}$, where $\mathcal{L} \subseteq \mathrm{Imp}(M)$ is a set of valid implications of $\mathbb{K}$. The set of all valid implications of $\mathbb{K}$ is called the *(implicational) theory* of $\mathbb{K}$, and is denoted by $\mathrm{Th}(\mathbb{K})$.♢

Note that the condition $A' \subseteq B'$ on the validity of an implication $A \to B$ in $\mathbb{K}$ can be characterized as saying that every object which has all attributes from A has all attributes from B. This coincides with our initial example.

Moreover, the condition $A' \subseteq B'$ can equally be rephrased as

$$B \subseteq A'', \tag{2.7}$$

i. e. all elements of the conclusion of $A \to B$ are in the closure of the premise with respect to the closure operator $(\cdot)''$. This little observation can be helpful for proofs.

2.3.2 Example As already mentioned, the implication $\{\,\mathsf{StarFleet}\,\} \to \{\,\mathsf{Honorable}\,\}$ holds in $\mathbb{K}_{\mathsf{TNG}}$. On the other hand, the implication $\{\,\mathsf{Artificial}\,\} \to \{\,\mathsf{Human}\,\}$ is not valid in $\mathbb{K}_{\mathsf{TNG}}$. ♢

Implications introduce a flavor of logic into formal concept analysis (indeed, implications can be seen as a notational variant of definite Horn formulas of the propositional variables M). As such, we can talk about *entailment* between implications in a very natural sense.

2.3.3 Example In $\mathbb{K}_{\mathsf{TNG}}$ the implications

$$\begin{aligned} \{\,\mathsf{Human}\,\} &\to \{\,\mathsf{Honorable}\,\} \\ \{\,\mathsf{Honorable}\,\} &\to \{\,\mathsf{StarFleet}\,\} \\ \{\,\mathsf{Human}\,\} &\to \{\,\mathsf{StarFleet}\,\} \end{aligned}$$

are all valid. However, we can intuitively see that the implication $\{\,\mathsf{Human}\,\} \to \{\,\mathsf{StarFleet}\,\}$ is entailed by the other two: by the first implication, every object that has Human as attribute has also Honorable as attribute. By the second implication, every object that has Honorable as attribute also has StarFleet as attribute. Therefore, every object that has Human as attribute also has StarFleet as attribute. ♢

We shall put this more formally in the following definition.

2.3.4 Definition (Entailment between Implications) Let M be a set and let $\mathcal{L} \subseteq \mathrm{Imp}(M)$. Then an implication $(A \to B) \in \mathrm{Imp}(M)$ is *entailed by* $\mathcal{L}$, written $\mathcal{L} \models (A \to B)$, if and only if for all formal contexts $\mathbb{K}$ with attribute set M it is true that

$$\mathbb{K} \models \mathcal{L} \implies \mathbb{K} \models (A \to B).$$

We shall denote with $\mathrm{Cn}_M(\mathcal{L})$ the set of all implications on M which follow from $\mathcal{L}$. We may drop the subscript M, and may only write $\mathrm{Cn}(\mathcal{L})$, if it is clear from the context.

If $\mathcal{K} \subseteq \mathrm{Imp}(M)$, then we say that $\mathcal{L}$ and $\mathcal{K}$ are *equivalent* if and only if $\mathrm{Cn}_M(\mathcal{L}) = \mathrm{Cn}_M(\mathcal{K})$. ♢

The mapping $\mathrm{Cn}_M \colon \mathfrak{P}(\mathrm{Imp}(M)) \to \mathfrak{P}(\mathrm{Imp}(M))$ is a first example of a closure operator which arises due to implications. However, there is also another closure operator induced by a set $\mathcal{L}$ of implications, which lets us easily decide whether an implication $A \to B$ follows from $\mathcal{L}$ or not.

2.3.5 Definition (Induced Closure Operator) Let M be a set and let $\mathcal{L} \subseteq \mathrm{Imp}(M)$. Define for $X \subseteq M$

$$\begin{aligned} \mathcal{L}^1(X) &:= X \cup \bigcup \{\, B \mid (A \to B) \in \mathcal{L}, A \subseteq X \,\}, \\ \mathcal{L}^{i+1} &:= \mathcal{L}^1(\mathcal{L}^i(X)) \quad (i \in \mathbb{N}_{>0}). \end{aligned}$$

Then the *induced closure operator* of $\mathcal{L}$ is defined by the mapping $X \mapsto \mathcal{L}(X)$, where

$$\mathcal{L}(X) := \bigcup_{i \in \mathbb{N}_{>0}} \mathcal{L}^i(X).$$

If no confusion is possible, we shall denote the closure operator induced by $\mathcal{L}$ again with $\mathcal{L}$, i. e. we shall identify the set of implications and its induced closure operator. ♢

It is easy to see that the induced closure operator is indeed a closure operator on $(\mathfrak{P}(M), \subseteq)$. Furthermore, the induced closure operator yields a characterization of entailment between implications. For this we first consider some technical results.

2.3.6 Proposition *Let $\mathbb{K} = (G, M, I)$ be a formal context and let $\mathcal{L} \subseteq \mathrm{Th}(\mathbb{K})$ a set of valid implications of $\mathbb{K}$. Then*

$$\mathcal{L}(A) \subseteq A''.$$

Proof By Lemma 2.1.9, it suffices to show that $A' \subseteq \mathcal{L}(A)'$. Let $g \in A'$. If then $(X \to Y) \in \mathcal{L}$ is such that $X \subseteq A$, then

$$g \in A' \subseteq X' \subseteq Y',$$

since $X \to Y$ holds in $\mathbb{K}$. Therefore,

$$g \in A' \cap \bigcap \{\, Y' \mid (X \to Y) \in \mathcal{L}, X \subseteq A \,\}.$$

However,

$$\begin{aligned} & A' \cap \bigcap \{\, Y' \mid (X \to Y) \in \mathcal{L}, X \subseteq A \,\} \\ = {} & (A \cup \bigcup \{\, Y \mid (X \to Y) \in \mathcal{L}, X \subseteq A \,\})' \\ = {} & (\mathcal{L}^1(A))' \end{aligned}$$

by Lemma 2.1.10, and thus $g \in (\mathcal{L}^1(A))'$. Iterating this argumentation yields $g \in (\mathcal{L}^i(A))'$ for all $i \in \mathbb{N}_{>0}$, and therefore

$$g \in \bigcap_{i \in \mathbb{N}_{>0}} (\mathcal{L}^i(A))' = \Big(\bigcup_{i \in \mathbb{N}_{>0}} \mathcal{L}^i(A) \Big)' = (\mathcal{L}(A))'$$

as required. □

2.3.7 Proposition *Let $\mathcal{L} \subseteq \mathrm{Imp}(M)$ for some set M. Then the formal context*

$$\mathbb{K}_{\mathcal{L}} := (\{\, \mathcal{L}(A) \mid A \subseteq M \,\}, M, \ni)$$

satisfies $X'' = \mathcal{L}(X)$ for all $X \subseteq M$.

Note that this proposition has the interesting consequence that $\mathrm{Th}(\mathbb{K}_{\mathcal{L}}) = \mathrm{Cn}_M(\mathcal{L})$. In other words, every set $\mathcal{L}$ of implications which is closed under Cn_M can be represented as a theory of a suitable chosen formal context, namely $\mathbb{K}_{\mathcal{L}}$.

Proof It is easy to see that $\mathcal{K}_{\mathcal{L}} \models \mathcal{L}$: if $(X \to Y) \in \mathcal{L}$ and $g \in X'$, then $\{ g \}' \supseteq X$. However, every set $\{ g \}'$ is of the form $\mathcal{L}(A)$ for some $A \subseteq M$. Therefore, we have $\mathcal{L}(A) \supseteq X$. Applying $\mathcal{L}$ on both sides yields $\mathcal{L}(A) \supseteq \mathcal{L}(X)$. Since $(X \to Y) \in \mathcal{L}$, $Y \subseteq \mathcal{L}(X)$ and therefore $\{ g \}' \supseteq \mathcal{L}(X) \supseteq Y$, so $g \in Y'$ as required.

For the other direction $\mathcal{L}(X) \supseteq X''$ we first show $(\mathcal{L}(X))'' = \mathcal{L}(X)$. Then $X \subseteq \mathcal{L}(X)$ implies

$$X'' \subseteq (\mathcal{L}(X))'' = \mathcal{L}(X).$$

We compute

$$(\mathcal{L}(X))' = \{ \mathcal{L}(A) \mid A \subseteq M, \mathcal{L}(A) \supseteq \mathcal{L}(X) \},$$

and obtain in particular that $\mathcal{L}(X) \in (\mathcal{L}(X))'$. On the other hand, for a subset $\mathcal{A}$ of objects of $\mathbb{K}_{\mathcal{L}}$ we have

$$\mathcal{A}' = \bigcap_{A \in \mathcal{A}} A,$$

since the incidence relation of $\mathbb{K}_{\mathcal{L}}$ is just $\ni$. Therefore,

$$\begin{aligned}(\mathcal{L}(X))'' &= \{ \mathcal{L}(A) \mid A \subseteq M, \mathcal{L}(A) \supseteq \mathcal{L}(X) \}' \\ &= \bigcap \{ \mathcal{L}(A) \mid A \subseteq M, \mathcal{L}(A) \supseteq \mathcal{L}(X) \} \\ &= \mathcal{L}(X),\end{aligned}$$

as required. □

Based on these two technical results we can now rephrase entailment between implications in terms of induced closure operators.

2.3.8 Lemma *Let M be a set and let $\mathcal{L} \subseteq \mathrm{Imp}(M)$. Then for $(A \to B) \in \mathcal{L}$ it is true that*

$$\mathcal{L} \models (A \to B) \iff B \subseteq \mathcal{L}(A). \tag{2.8}$$

Proof Suppose that $\mathcal{L} \models (A \to B)$. For the formal context $\mathbb{K}_{\mathcal{L}}$ from Proposition 2.3.7 we have

$$\mathcal{L}(X) = X''$$

for all $X \subseteq M$. Since $\mathbb{K}_{\mathcal{L}} \models (A \to B)$, $A' \subseteq B'$ is true in $\mathbb{K}_{\mathcal{L}}$. But then

$$B \subseteq A'' = \mathcal{L}(A)$$

as required.

Conversely, let $B \subseteq \mathcal{L}(A)$, and let $\mathbb{K}$ be a formal context such that $\mathbb{K} \models \mathcal{L}$. Then $\mathcal{L}(A) \subseteq A''$ by Proposition 2.3.6, and therefore $B \subseteq \mathcal{L}(A) \subseteq A''$. But then $A' \subseteq B'$, i. e. $A \to B$ holds in $\mathbb{K}$. Since $\mathbb{K}$ was chosen arbitrarily, it follows that $\mathcal{L} \models (A \to B)$. □

The computation of $\mathcal{L}(A)$ for sets $A \subseteq M$ can be done naively with time quadratic in $|\mathcal{L}|$, assuming the size of the underlying set M is fixed. However, one can improve this by using the "LinClosure" algorithm from the theory of relational databases [70], which achieves the same goal with time linear in $|\mathcal{L}|$. Finally, one can also exploit the tight connection to Horn-formulas as mentioned above, and use the algorithm by Dowling and Gallier to decide satisfiability of Horn formulas in linear time [42].

2.4. Bases of Implications

When studying the valid implications of a formal context, it may be helpful not to consider the whole set, but a smaller still equivalent set of implications. The advantage of this approach is that this smaller set may be much easier to handle, especially from a computational point of view. We therefore introduce in this section the notion of a *base* of a set of implications, and introduce a well known minimal base of all valid implications of a formal context, the *canonical base*.

2.4.1 Definition (Sound and Complete Sets of Implications, Bases) Let $\mathcal{L} \subseteq \mathrm{Imp}(M)$ be a set of implications on a set M. A set $\mathcal{K} \subseteq \mathrm{Imp}(M)$ of implications on M is called *sound* for $\mathcal{L}$ if $\mathrm{Cn}_M(\mathcal{K}) \subseteq \mathrm{Cn}_M(\mathcal{L})$. The set $\mathcal{K}$ is called *complete* for $\mathcal{L}$ if $\mathrm{Cn}_M(\mathcal{K}) \supseteq \mathrm{Cn}_M(\mathcal{L})$. Finally, the set $\mathcal{K}$ is called a *base* of $\mathcal{L}$ if $\mathcal{K}$ is sound and complete for $\mathcal{L}$, i. e. if

$$\mathrm{Cn}_M(\mathcal{K}) = \mathrm{Cn}_M(\mathcal{L}). \tag{2.9}$$

If $\mathcal{L} = \mathrm{Th}(\mathbb{K})$, then a set $\mathcal{K}$ which is sound or complete for $\mathcal{L}$ is also called sound or complete for $\mathbb{K}$, respectively. Moreover, we shall call bases of $\mathcal{L}$ also bases of $\mathbb{K}$.

A base $\mathcal{K}$ of $\mathcal{L}$ is called *irredundant* if no proper subset of $\mathcal{K}$ is a base of $\mathcal{L}$. $\mathcal{K}$ is called *minimal*, if there does not exist a base of $\mathcal{L}$ of smaller cardinality. ♢

Note that any base of a set $\mathcal{L}$ is also a base of $\mathrm{Cn}_M(\mathcal{L})$, and vice versa.

A simple example of a base of a formal context $\mathbb{K} = (G, M, I)$ is the set

$$\mathcal{K} = \{\, A \to A'' \mid A \subseteq M, A \neq A'' \,\}. \tag{2.10}$$

Obviously, this set contains only valid implications of $\mathbb{K}$. Furthermore, if $A \to B$ holds in $\mathbb{K}$, then $B \subseteq A''$, and by Lemma 2.3.8, $A \to B$ follows from $\{\, A \to A'' \,\}$, and thus from $\mathcal{K}$. From this we can also infer that[1]

$$\mathrm{Th}(\mathbb{K})(A) = A''$$

for all $A \subseteq M$, because

$$\mathrm{Th}(\mathbb{K})(A) = \mathcal{K}(A) = A''.$$

[1] This notation $\mathrm{Th}(\mathbb{K})(A)$ may be a bit misleading. What is meant here is that the induced closure operator of the set $\mathrm{Th}(\mathbb{K})$ of implications is applied to A, so this expression could also be written as $(\mathrm{Th}(\mathbb{K}))(A)$.

To see the latter equality we first observe that $\mathcal{K}(A) \supseteq A''$, because $(A \to A'') \in \mathcal{K}$. On the other hand, A'' is closed under $\mathcal{K}$ and is a superset of A, thus $\mathcal{K}(A) \subseteq A''$, and so equality holds.

Checking soundness of a set of implications $\mathcal{L} \subseteq \mathrm{Imp}(M)$ for a formal context $\mathbb{K}$ is rather trivial, but checking completeness of $\mathcal{L}$ for $\mathbb{K}$ is not so easy (indeed, it is coNP-complete [63, Theorem 12]). There is a simple characterization, however, if $\mathcal{L}$ is complete for a formal context, which is at least helpful for proofs.

2.4.2 Lemma *Let $\mathbb{K} = (G, M, I)$ be a formal context and let $\mathcal{L} \subseteq \mathrm{Imp}(M)$. Then $\mathcal{L}$ is complete for $\mathbb{K}$ if and only if*

$$\forall U \subseteq M\colon \mathcal{L}(U) = U \implies U = U''. \tag{2.11}$$

Proof Assume that $\mathcal{L}$ is complete for $\mathbb{K}$ and suppose by that there exists $U \subseteq M$ such that $U \neq U''$. Then the implication $U \to U''$ is valid in $\mathbb{K}$, and since $\mathcal{L}$ is complete for $\mathbb{K}$ we obtain that

$$\mathcal{L} \models (U \to U'').$$

But then $U'' \subseteq \mathcal{L}(U)$, and thus $U \subsetneq \mathcal{L}(U)$, and in particular $U \neq \mathcal{L}(U)$ as required.

Now suppose that (2.11) holds, and let $U \subseteq M$. Since $\mathcal{L}(U)$ is closed under $\mathcal{L}$, i. e. $\mathcal{L}(\mathcal{L}(U)) = \mathcal{L}(U)$, we obtain from (2.11) that

$$\mathcal{L}(U) = (\mathcal{L}(U))''$$

for each $U \subseteq M$. But then $U'' \subseteq (\mathcal{L}(U))'' = \mathcal{L}(U)$, and therefore $\mathcal{L} \models (U \to U'')$ by Lemma 2.3.8. Therefore, $\mathcal{L}$ is complete for $\mathbb{K}$. □

The base from (2.10) is not very practical, as it will almost always have exponentially many elements in the size of the set $|M|$ of attribute of $\mathbb{K}$. Processing such a base may be computationally infeasible, and it is therefore desirable to have a smaller base. Indeed, it is possible to explicitly describe even a *minimal* base of every formal context $\mathbb{K}$ (or for every set of implications), namely its *canonical base* [51, 70] (also called *Duquenne-Guiges base* or *stem base*).

To introduce this base, we first need to discuss the notion of *pseudo-intents*. Note that we only introduce the canonical base for bases of formal contexts. However, this can be done without loss of generality, as a base of a set $\mathcal{L}$ of implications is always also a base of its closure $\mathrm{Cn}_M(\mathcal{L})$, and such sets can be represented as theories of formal contexts by virtue of Proposition 2.3.7.

The variant of pseudo-intents as introduced here is due to [92].

2.4.3 Definition (Pseudo-Intents) Let M be a set, $\mathbb{K}$ a formal context with attribute set M and let $\mathcal{S}$ be a set of implications on M. Then a set $P \subseteq M$ is called an *$\mathcal{S}$-pseudo-intent* of $\mathbb{K}$ if and only if

i. $P \neq P''$,

ii. $P = \mathcal{S}(P)$, and

iii. for each $\mathcal{S}$-pseudo-intent Q of $\mathbb{K}$ satisfying $Q \subsetneq P$, it is true that $Q'' \subseteq P$.

If $\mathcal{S} = \varnothing$, then an $\mathcal{S}$-pseudo-intent is just called a *pseudo-intent*. ♢

As it is given, the definition of $\mathcal{S}$-pseudo-intents is quite inaccessible, and the motivation for it may only become apparent while working with them. However, the role of the set $\mathcal{S}$ can already be motivated now: this set will be used as *background knowledge* when computing the canonical base. Let us make this more precise, and recall that we have defined the notion of a base $\mathcal{K}$ of a set of implications $\mathcal{L}$ to just mean that $\mathrm{Cn}_M(\mathcal{K}) = \mathrm{Cn}_M(\mathcal{L})$. The motivation for this was that we want bases of the set $\mathcal{L}$ to be a different, potentially smaller but logically equivalent representation of $\mathcal{L}$. However, within this scenario, we can assume that we already "know" a certain set $\mathcal{S} \subseteq \mathcal{L}$ of implications, and we only want to find a base that somehow represents the "difference" between $\mathcal{L}$ and $\mathcal{S}$.

2.4.4 Definition (Bases with Implicational Background Knowledge) Let $\mathcal{L}$ and $\mathcal{S}$ be sets of implications on a set M such that $\mathcal{S} \subseteq \mathrm{Cn}_M(\mathcal{L})$. A *base* of $\mathcal{L}$ *with background knowledge* $\mathcal{S}$ is a set $\mathcal{K}$ of implications on M such that

$$\mathrm{Cn}_M(\mathcal{K} \cup \mathcal{S}) = \mathrm{Cn}_M(\mathcal{L}).$$

The notions of *irredundancy* and *minimality* of bases with background knowledge are defined analogously to Definition 2.4.1: $\mathcal{K}$ is an *irredundant* base of $\mathcal{L}$ with background knowledge $\mathcal{S}$ if and only if no proper subset of $\mathcal{K}$ is a base of $\mathcal{L}$ with background knowledge $\mathcal{S}$. $\mathcal{K}$ is a *minimal* base of $\mathcal{L}$ with background knowledge $\mathcal{S}$ if and only if $\mathcal{K}$ has minimal cardinality among all bases of $\mathcal{L}$ with background knowledge $\mathcal{S}$. ♢

The canonical base, which we shall introduce shortly, can be defined such that it also allows for background knowledge. This background knowledge then will play the role of the set $\mathcal{S}$ in the definition of an $\mathcal{S}$-pseudo-intent. Moreover, it can be shown that the canonical base is a minimal base with the given background knowledge.

2.4.5 Definition (Canonical Base) Let $\mathbb{K} = (G, M, I)$ be a formal context and let $\mathcal{S} \subseteq \mathrm{Imp}(M)$. Then the *canonical base* $\mathrm{Can}(\mathbb{K}, \mathcal{S})$ of $\mathbb{K}$ *with background knowledge* $\mathcal{S}$ is defined as

$$\mathrm{Can}(\mathbb{K}, \mathcal{S}) := \{\, P \to P'' \mid P \text{ is an } \mathcal{S}\text{-pseudo-intent of } \mathbb{K} \,\}.$$

If $\mathcal{S} = \varnothing$, then we just write $\mathrm{Can}(\mathbb{K})$ for $\mathrm{Can}(\mathbb{K}, \mathcal{S})$. ♢

The classical result about the canonical base can now be stated as follows.

2.4.6 Theorem *Let* $\mathbb{K}$ *be a finite formal context and let* $\mathcal{S} \subseteq \mathrm{Th}(\mathbb{K})$. *Then* $\mathrm{Can}(\mathbb{K}, \mathcal{S})$ *is a minimal base of* $\mathbb{K}$ *with background knowledge* $\mathcal{S}$.

This formulation assumes that the background knowledge $\mathcal{S}$ is sound for $\mathbb{K}$. However, this is not only not necessary, but we shall also later encounter situations where our background knowledge is not valid, but where the corresponding canonical base has still a meaningful application. We therefore slightly generalize Theorem 2.4.6 to yield the following theorem.

2.4.7 Theorem *Let* $\mathbb{K} = (G, M, I)$ *be a formal context and let* $\mathcal{S} \subseteq \mathrm{Imp}(M)$. *Then* $\mathrm{Can}(\mathbb{K}, \mathcal{S})$ *is a set of valid implications of* $\mathbb{K}$ *such that* $\mathrm{Can}(\mathbb{K}, \mathcal{S}) \cup \mathcal{S}$ *is complete for* $\mathbb{K}$. *Moreover,* $\mathrm{Can}(\mathbb{K}, \mathcal{S})$ *has minimal cardinality among all sets of valid implications satisfying this property.*

The proof about the minimal cardinality of $\mathrm{Can}(\mathbb{K}, \mathcal{S})$ is a straight-forward adaption of the proof of [41, Theorem 3.8].

Proof We need to show the following three statements:

i. $\mathrm{Can}(\mathbb{K}, \mathcal{S})$ is sound for $\mathbb{K}$;

ii. $\mathrm{Can}(\mathbb{K}, \mathcal{S}) \cup \mathcal{S}$ is complete for $\mathbb{K}$ and

iii. $\mathrm{Can}(\mathbb{K}, \mathcal{S})$ has minimal cardinality among all sets $\mathcal{P} \subseteq \mathrm{Th}(\mathbb{K})$ of implications such that $\mathcal{P} \cup \mathcal{S}$ is complete for $\mathbb{K}$.

For i we just note that $\mathrm{Can}(\mathbb{K}, \mathcal{S})$ only consists of implications which are of the form $P \to P''$, which are of course valid in $\mathbb{K}$.

For the completeness as claimed in ii we shall make use of Lemma 2.4.2 by showing that every closed set of $\mathrm{Can}(\mathbb{K}, \mathcal{S}) \cup \mathcal{S}$ is already an intent of $\mathbb{K}$. For readability, let us set $\mathcal{L} := \mathrm{Can}(\mathbb{K}, \mathcal{S}) \cup \mathcal{S}$.

So let $U \subseteq M$ be such that $\mathcal{L}(U) = U$. If then $V \subsetneq U$ is an $\mathcal{S}$-pseudo-intent of $\mathbb{K}$, then $V'' \subseteq U$, since $(V \to V'') \in \mathcal{L}$. Furthermore, $\mathcal{S}(U) = U$ because $\mathcal{S} \subseteq \mathcal{L}$. Hence, if we assume by contradiction that $U \neq U''$, then U would be an $\mathcal{S}$-pseudo-intent of $\mathbb{K}$, i. e. $(U \to U'') \in \mathrm{Can}(\mathbb{K}, \mathcal{S}) \subseteq \mathcal{L}$. But then $U'' \subseteq \mathcal{L}(U) = U$, i. e. $U = U''$ contradicting $U \neq U''$. Therefore, $U = U''$ and since U was chosen arbitrarily, $\mathcal{L}$ is complete by Lemma 2.4.2.

For the last claim iii let $\mathcal{P}$ be another set of valid implications of $\mathbb{K}$ such that $\mathcal{P} \cup \mathcal{S}$ is complete for $\mathbb{K}$. Without loss of generality we may assume that $\mathcal{P}$ only contains implications of the form $U \to U''$ for suitable $U \subseteq M$.

To prove iii we shall now show that for each $\mathcal{S}$-pseudo-intent P of $\mathbb{K}$ there exists a set $U_P \subseteq M$ such that $(U_P \to U_P'') \in \mathcal{P}$, and that in addition the mapping $P \to U_P$ is injective. From this it immediately follows that $|\mathcal{P}| \geqslant |\mathrm{Can}(\mathbb{K}, \mathcal{S})|$.

So let P be an $\mathcal{S}$-pseudo-intent of $\mathbb{K}$. Then $P \neq P''$. As $\mathcal{P} \cup \mathcal{S}$ is complete for $\mathbb{K}$ and $\mathcal{S}(P) = P$, there exists an implication $(X \to X'') \in \mathcal{P}$ such that $X \subseteq P$ and $X'' \nsubseteq P$. Set $U_P := X$.

To see that the resulting map $P \to U_P$ is indeed injective, let P and Q be two $\mathcal{S}$-pseudo-intents of $\mathbb{K}$, and assume that $U_P = U_Q =: U$. Then $U \subseteq P$ and $U \subseteq Q$ by definition of U_P and U_Q. Hence $U \subseteq P \cap Q$, and therefore $U'' \subseteq (P \cap Q)''$.

Then the fact that $U'' = U_P'' \nsubseteq P$ and $U'' \subseteq (P \cap Q)''$ implies that $(P \cap Q)'' \nsubseteq P$. Therefore, $(P \cap Q)'' \nsubseteq P \cap Q$, and in particular

$$(P \cap Q)'' \neq P \cap Q. \tag{2.12}$$

Recall that $\mathcal{S}(P) = P$ and $\mathcal{S}(Q) = Q$. This implies that $\mathcal{S}(P \cap Q) = P \cap Q$, since the intersection of closed sets is again closed. But then, by Equation (2.12) and the fact that $\mathrm{Can}(\mathbb{K}, \mathcal{S}) \cup \mathcal{S}$ is complete for $\mathbb{K}$, there must exist an implication $(R \to R'') \in \mathrm{Can}(\mathbb{K}, \mathcal{S})$ such that

$$R \subseteq P \cap Q \quad \text{and} \quad R'' \nsubseteq P \cap Q.$$

Without loss of generality we assume that $R'' \nsubseteq Q$. But then $R \subseteq Q$ is an $\mathcal{S}$-pseudo-intent of $\mathbb{K}$, so if $R \subsetneq Q$, it must be that $R'' \subseteq Q$, which is not the case. Therefore, $R = Q$. Since $R \subseteq P \cap Q$, we obtain $Q \subseteq P \cap Q$, i. e. $Q = P \cap Q$ and therefore

$$Q \subseteq P \tag{2.13}$$

Since $(P \cap Q)'' \nsubseteq P$, Equation (2.13) implies that $Q'' \nsubseteq P$. Since $Q \subseteq P$, the $\mathcal{S}$-pseudo-intents Q and P cannot be different, so we obtain $P = Q$.

Therefore, the mapping $P \mapsto U_P$ is injective and we have proven iii. □

Theorem 2.4.6 is now an immediate consequence of Theorem 2.4.7.

The minimality properties of the canonical base makes it particularly interesting for practical applications. Indeed, besides these minimality properties discussed above, the canonical base also allows for a comparably simple computation. On the other hand, the minimality of the canonical base does not save us from exponentially big bases.

2.4.8 Example The following example is taken from [66]. Let $n \in \mathbb{N}$ and let us consider the formal context $\mathbb{K}_n$ as shown in Figure 2.2. The object set $G = G_1 \cup G_2$ of $\mathbb{K}$ is defined as

$$G_1 = \{ g_1, \ldots, g_n \}$$
$$G_2 = \{ \overline{g}_1, \ldots, \overline{g}_{2n} \},$$

where all $g_i, \overline{g}_i$ are distinct, and the attribute set $M = M_1 \cup M_2 \cup \{ \overline{m}_0 \}$ is given by

$$M_1 = \{ m_1, \ldots, m_n \},$$
$$M_2 = \{ \overline{m}_1, \ldots, \overline{m}_n \},$$

where again all $m_i, \overline{m}_i$ are distinct.

$\mathbb{K}_n$	m_0	$m_1 \dots m_n$	$\overline{m}_1, \dots \overline{m}_n$
g_1 $\vdots$ g_n		I_1	I_2
$\overline{g}_1$ $\vdots$ $\overline{g}_{2n}$	$\times$ $\times$ $\times$	I_3	

Figure 2.2.: A Formal Context with Exponentially Many Pseudo-Intents

The relations I_1, I_2, I_3 are essentially $\neq$, more precisely

$$
\begin{aligned}
(g_i, m_j) \in I_1 &:\Longleftrightarrow i \neq j \\
(g_i, \overline{m}_j) \in I_2 &:\Longleftrightarrow i \neq j \\
(\overline{g}_i, m_j) \in I_3 &:\Longleftrightarrow i \neq j \\
(\overline{g}_i, \overline{m}_j) \in I_3 &:\Longleftrightarrow i \neq j + n
\end{aligned}
$$

Then the claim is that the number of pseudo-intents of $\mathbb{K}_n$ is at least 2^n. To see this we first observe that the set $\{ m_1, \dots, m_n \}$ is a pseudo-intent of $\mathbb{K}_n$. This is because if

$$B = \{ m_{j_1}, \dots, m_{j_k} \} \subseteq \{ m_1, \dots, m_n \},$$

then

$$B' = (G_1 \backslash \{ g_{j_1}, \dots, g_{j_k} \}) \cup (G_2 \backslash \{ \overline{g}_{j_1}, \dots, \overline{g}_{j_k} \})$$

and then $B = B''$. On the other hand, the set $\{ m_1, \dots, m_n \}$ is not closed, since

$$\{ m_1, \dots, m_n \}'' = \{ \overline{g}_{n+1}, \dots, \overline{g}_{2n} \}' = \{ m_0, m_1, \dots, m_n \}.$$

Therefore, as all subsets of $\{ m_1, \dots, m_n \}$ are closed and the set itself is not, it is a pseudo-intent of $\mathbb{K}_n$ by definition.

Now, if we replace an attribute m_i with $\overline{m}_i$, the resulting set $\{ m_1, \dots, m_{i-1}, \overline{m}_i, m_{i+1}, \dots, m_n \}$ is still a pseudo-intent of $\mathbb{K}_n$, using a similar argument: the set $\{ m_1, \dots, \overline{m}_i, \dots, m_n \}$ is not closed, since

$$\{ m_1, \dots, \overline{m}_i, \dots, m_n \}'' = \{ m_0, m_1, \dots, \overline{m}_i, \dots, m_n \},$$

and every subset $C \subseteq \{ m_1, \dots, \overline{m}_i, \dots, m_n \}$ is closed, by the same arguments as we used for B.

Therefore, $\mathbb{K}_n$ has at least 2^n pseudo-intents and thus

$$|\mathrm{Can}(\mathbb{K}_n)| \geqslant 2^n. \qquad \diamondsuit$$

In spite of this rather disappointing result it may still pay off in practical applications to compute the canonical base, as its size may still be considerably smaller then the size of the base from (2.10). In the following, we shall discuss a standard algorithm that computes the canonical base of a formal context.

The first, rather technical definition we need to consider is the one of the *lectic order* on the powerset $\mathfrak{P}(M)$ of a linearly ordered set $(M, \leqslant_M)$, i. e. an ordered set $(M, \leqslant_M)$ where for any two elements $x, y \in M$ it is true that $x \leqslant_M y$ or $y \leqslant_M x$.

2.4.9 Definition (Lectic Order) Let $(M, \leqslant_M)$ be a linearly ordered set, and let $i \in M$. Then for two sets $A, B \subseteq M$, A is called *lectically smaller* than B *at position* i, written $A \prec_i B$, if

$$A \prec_i B \;:\Longleftrightarrow\; i = \min_{\leqslant_M}(A \triangle B) \text{ and } i \in B,$$

where

$$A \triangle B := (A \backslash B) \cup (B \backslash A)$$

is the symmetric difference of A and B.

Then the *lectic order* on $\mathfrak{P}(M)$ induced by $\leqslant_M$ is the relation $\preceq$ defined as

$$A \preceq B \;:\Longleftrightarrow\; A = B \text{ or } A \prec_i B \text{ for some } i \in M. \qquad \diamondsuit$$

The fact that $A \prec_i B$ can be understood as the statement that the smallest element (with respect to $\leqslant_M$) in which A and B differ belongs to B.

2.4.10 Example Let $M = \{0, 1, 2\}$ and let $\leqslant_M$ be given by $2 \leqslant_M 1 \leqslant_M 0$. Then

$$\{0\} \prec_i \{1\}.$$

Moreover, all subsets of M are ordered by $\preceq$ as follows

$$\varnothing \preceq \{0\} \preceq \{1\} \preceq \{0,1\} \preceq \{2\} \preceq \{0,2\} \preceq \{0,1,2\}.$$

If we encode a subset of M as a binary number like we did in Example 2.1.2, so for example $\{0, 2\}$ would be 101 and $\{0\}$ would be 001, then the above lectic order can also be written as

$$000 \preceq 001 \preceq 010 \preceq 011 \preceq 100 \preceq 101 \preceq 111$$

which resembles the usual linear order on binary numbers. $\diamondsuit$

It is not hard to see (but technical to prove) that every lectic order is indeed a linear order on $\mathfrak{P}(M)$, i. e. the ordered set $(\mathfrak{P}(M), \preceq)$ is a linearly ordered set. Moreover the lectic order *extends* the usual subset-order on $\mathfrak{P}(M)$, in the sense that for all $A, B \subseteq M$ it is true that

$$A \subseteq B \implies A \preceq B,$$

irrespective of the linear order $\leqslant_M$ used to define $\preceq$.

Let now c be a closure operator on the linearly ordered set $(M, \leqslant_M)$. In the following we shall introduce the Next-Closure algorithm [46, 48] that allows us to compute all closed sets of c ordered by the lectic order on $\mathfrak{P}(M)$ induced by $\leqslant_M$. Moreover, this algorithm has the advantage of usually being much faster than just applying the closure operator c to all subsets of M and collecting the results.

However, before we shall discuss the algorithm, let us first introduce an abbreviation. For a set $A \subseteq M$ and an element $i \in M$, let us write

$$A \oplus_c i := c(\{\, a \in A \mid a \leqslant_M i \,\} \cup \{\, i \,\}).$$

Then the following theorem holds, which is [48, Theorem 5].

2.4.11 Theorem *Let $(M, \leqslant_M)$ be a linearly ordered set, and let $\preceq$ be the lectic order on $\mathfrak{P}(M)$ induced by $\leqslant_M$. Furthermore, let c be a closure operator on $(M, \leqslant_M)$ and let $A \subseteq M$. Then, if there exists a set B such that $A \preceq B$ and B is closed under c, then*

$$\min_{\preceq} \{\, A \preceq B \mid B = c(B) \,\} = A \oplus_c i,$$

where i is $\leqslant_M$-maximal among all elements $j \in M$ satisfying $A \prec_j A \oplus_c j$, i. e.

$$i = \max_{\leqslant_M} \{\, j \in M \mid A \prec_j A \oplus_c j \,\}.$$

The proof of this theorem is rather technical, and we shall not repeat it here.

The main advantage of this theorem is now that it immediately gives rise to an effective algorithm to compute the lectically next closed set after a given one. Algorithm 1 shows an example implementation, which returns **nil** if no lectically next closed set exists.

The Next-Closure algorithm can in particular be used to compute all intents (and thus all formal concepts) of a formal context, because intents are just those subsets of the attribute sets which are closed under $(\cdot)''$. The following example illustrates this.

2.4.12 Example Let us again consider our Star Trek context $\mathbb{K}_{\mathsf{TNG}}$. Clearly, $\varnothing = \varnothing''$ is true in that context. For computing more intents using Next-Closure, let us order the attributes of this context as follows

$$\mathsf{Human} < \mathsf{Honorable} < \mathsf{Artificial} < \mathsf{StarFleet}.$$

Algorithm 1 (Next-Closure)

```
0 define next-closure(M, ⩽_M, A, c)
1   C := { i ∈ M | A ≺_i A ⊕_c i }
2   if C = ∅
3     return nil
4   else
5     return A ⊕_c max_⩽M(C)
6   end
7 end
```

We now want to compute the lectically next closed set of $(\cdot)''$ after $\varnothing$, i. e. the next intent of $\mathbb{K}_{\mathsf{TNG}}$ after $\varnothing$. For this, we compute

$$\varnothing \oplus_{(\cdot)''} \mathsf{StarFleet} = \{\, \mathsf{StarFleet} \,\}'' = \{\, \mathsf{Honorable}, \mathsf{StarFleet} \,\},$$

but it is not true that

$$\varnothing \prec_{\mathsf{StarFleet}} \{\, \mathsf{Honorable}, \mathsf{StarFleet} \,\}.$$

So we continue with Artificial and obtain

$$\varnothing \oplus_{(\cdot)''} \mathsf{Artificial} = \{\, \mathsf{Artificial} \,\}'' = \mathsf{Artificial}$$

and indeed $\varnothing \prec_{\mathsf{Artificial}} \{\, \mathsf{Artificial} \,\}$, so this set is the lectically next intent of $\mathbb{K}$ after $\varnothing$. The corresponding formal concept is

$$(\{\, \mathsf{Artificial} \,\}', \{\, \mathsf{Artificial} \,\}) = (\{\, \mathsf{Data}, \mathsf{BorgQueen} \,\}, \{\, \mathsf{Artificial} \,\}).$$

Continuing this process, we can successively compute all intents of $\mathbb{K}_{\mathsf{TNG}}$ this way, and thus also all formal concepts of $\mathbb{K}_{\mathsf{TNG}}$ as shown in Example 2.1.17. ♢

Based on the Next-Closure algorithm we can now discuss an algorithm that allows us to compute the canonical base of a formal context $\mathbb{K}$ with arbitrary background knowledge. Algorithm 2 gives an implementation of such an algorithm. We shall discuss the details of this algorithm when proving its correctness. Note that Algorithm 2 also contains the two auxiliary functions `next-closed-non-intent` and `first-closed-non-intent`, which are discussed below.

The correctness of `canonical-base` can now be stated as follows.

Algorithm 2 (Computing the Canonical Base with Background Knowledge)

```
0   define next-closed-non-intent (𝕂 = (G, M, I), ⩽_M, A, c)
1     ;; compute the lectically next non-intent of 𝕂 after A that is closed under c
2     P := next-closure(M, ⩽_M, A, c)
3     if P = nil then
4       return nil
5     else if P ≠ P'' then
6       return P
7     else
8       return next-closed-non-intent(𝕂, ⩽_M, P, c)
9     end
10  end
11
12  define first-closed-non-intent(𝕂 = (G, M, I), ⩽_M, c)
13    ;; compute the lectically first non-intent of 𝕂 that is closed under c
14    if ∅ ≠ ∅'' and c(∅) = ∅ then
15      return ∅
16    else
17      return next-closed-non-intent(𝕂, ⩽_M, ∅, c)
18    end
19  end
20
21  define canonical-base(𝕂 = (G, M, I), ⩽_M, 𝒮 ⊆ Imp(M))
22    ;; compute the canonical base of 𝕂 with background knowledge 𝒮
23    ;; in the lectic order induced by ⩽_M.
24
25    i := 0,
26    P_i := first-closed-non-intent(𝕂, ⩽_M, 𝒮),
27    ℒ_i := ∅
28
29    while P_i ≠ nil do
30      ℒ_{i+1} := ℒ_i ∪ { P_i → P_i'' },
31      P_{i+1} := next-closed-non-intent(𝕂, ⩽_M, P_i, ℒ_{i+1} ∪ 𝒮),
32      i := i + 1
33    end
34
35    return ℒ_i
36  end
```

2.4.13 Theorem *Let $\mathbb{K} = (G, M, I)$ be a finite formal context, i. e. both $|G|$ and $|M|$ are finite. Furthermore, let $\mathcal{K} \subseteq \mathrm{Imp}(M)$ and let $\leqslant_M$ be a linear order on M. Then the call to* `canonical-base` *terminates and it is then true that*

$$\mathrm{Can}(\mathbb{K}, \mathcal{K}) = \texttt{canonical-base}(\mathbb{K}, \leqslant_M, \mathcal{K}).$$

Before we are proving this theorem, we first consider the auxiliary functions also shown in Algorithm 2, and show that they yield what their names suggest.

2.4.14 Proposition *Let $\mathbb{K} = (G, M, I)$ be a finite formal context, $\leqslant_M$ a linear order on M and c a closure operator on $(M, \leqslant_M)$. Denote with $\preceq$ the lectic order on $\mathfrak{P}(M)$ induced by $\leqslant_M$.*

i. Let $A \subseteq M$ and define $S :=$ `next-closed-non-intent`$(\mathbb{K}, \leqslant_M, A, c)$*. Then*

$$S = \min_{\preceq} \{\, B \subseteq M \mid A \precneqq B, B = c(B), B \neq B'' \,\}$$

*if this minimum exists, and $S =$ **nil** otherwise.*

ii. Define $T :=$ `first-closed-non-intent`$(\mathbb{K}, \leqslant_M, c)$*. Then*

$$T = \min_{\preceq} \{\, B \subseteq M \mid B = c(B), B \neq B'' \,\}$$

*if this minimum exists, and $T =$ **nil** otherwise.*

Proof For the first statement observe that the algorithm considers in lectic order all sets $C \subseteq M$ where $A \precneqq C$ that are closed under c. Now, if a lectically smallest set $C \subseteq M$ with $A \precneqq C$ exists such that $C \neq C''$, then it is finally found by the algorithm and returned as the resulting value.

On the other hand, if no such set exists, then the variable P in the algorithm will eventually obtain the value M, and the subsequent iteration will return **nil**, since

$$\texttt{next-closure}(M, \leqslant_M, M, c) = \textbf{nil}.$$

For the second statement let us first assume that $\varnothing \neq \varnothing''$ and $c(\varnothing) = \varnothing$. Then clearly

$$\min_{\preceq} \{\, B \subseteq M \mid B = c(B), B \neq B'' \,\} = \varnothing = \texttt{first-closed-non-intent}(\mathbb{K}, \leqslant_M, c).$$

If $\varnothing = \varnothing''$ or $c(\varnothing) \neq \varnothing''$, then

$$\varnothing \precneqq \min_{\preceq} \{\, B \subseteq M \mid B = c(B), B \neq B'' \,\}.$$

Hence

$$\begin{aligned} \min_{\preceq} \{\, B \subseteq M \mid B = c(B), B \neq B'' \,\} &= \min_{\preceq} \{\, B \subseteq M, \varnothing \precneqq B, B = c(B), B \neq B'' \,\} \\ &= \texttt{next-closed-non-intent}(\mathbb{K}, \leqslant_M, \varnothing, c) \\ &= \texttt{first-closed-non-intent}(\mathbb{K}, \leqslant_M, c) \end{aligned}$$

as required. □

We shall now prove 2.4.13. Note that the proof itself is well-known, but we shall give it here nevertheless for the sake of completeness, and to convey some intuition why the implementation of `canonical-base` indeed computes the canonical base.

The main line of argumentation of the proof is as follows: using induction, we shall prove that in a call of `canonical-base`$(\mathbb{K}, \leqslant_M, \mathcal{S})$ for each i the following is true: if there exists an $\mathcal{S}$-pseudo-intent that is not within the list $P_0, \ldots, P_{i-1}$, then $P_i \neq$ **nil** and $P_0, \ldots, P_i$ are the lectically first $\mathcal{S}$-pseudo-intents of $\mathbb{K}$. Thus, if the algorithm finishes in iteration n, say, then $\mathcal{L}_n = \mathrm{Can}(\mathbb{K}, \mathcal{S})$, which is the value returned by the function.

Proof (Theorem 2.4.13) As before, denote with $\preceq$ the lectic order on $\mathfrak{P}(M)$ induced by $\leqslant_M$. As already mentioned, we shall prove by induction over the number i of iterations that if $\mathbb{K}$ has more than i $\mathcal{S}$-pseudo-intents, that then $P_i \neq$ **nil** and the sets

$$P_0, \ldots, P_i$$

are the first $i+1$ $\mathcal{S}$-pseudo-intents of $\mathbb{K}$ with respect to $\preceq$.

Let $i = 0$ and suppose that $\mathbb{K}$ has $\mathcal{S}$-pseudo-intents. If $\overline{P}_0$ is then the lectically first $\mathcal{S}$-pseudo-intent of $\mathbb{K}$, it is true that $\overline{P}_0$ is closed under $\mathcal{S}$ but not an intent of $\mathbb{K}$. Therefore, $P_0 \preceq \overline{P}_0$.

On the other hand, P_0 is already an $\mathcal{S}$-pseudo-intent of $\mathbb{K}$, because $P_0 \neq P_0''$, $\mathcal{S}(P_0) = P_0$ and if $Q \subsetneq P_0$ such that $\mathcal{S}(Q) = Q$, then $Q = Q''$ already holds, i. e. there are no $\mathcal{S}$-pseudo-intents strictly contained in P_0. Thus, $\overline{P}_0 \preceq P_0$ and therefore $P_0 = \overline{P}_0$, i. e. P_0 is the lectically first $\mathcal{S}$-pseudo-intent of $\mathbb{K}$. In particular, $P_0 \neq$ **nil**.

Now suppose $i > 0$ and suppose that $\mathbb{K}$ has more than i $\mathcal{S}$-pseudo-intents. By induction hypothesis, the list

$$P_0, \ldots, P_{i-1}$$

consists of the lectically first i $\mathcal{S}$-pseudo-intents. Let P be the lectically next $\mathcal{S}$-pseudo-intent after P_{i-1}. Then P is closed under $\mathcal{S}$ and not an intent of $\mathbb{K}$. In particular, $P_i \preceq P$, and $P_i \neq$ **nil**.

Furthermore, P_i is an $\mathcal{S}$-pseudo-intent of $\mathbb{K}$. To see this we first observe that $\mathcal{S}(P_i) = P_i$ and $P_i \neq P_i''$, so it suffices to show that for each $\mathcal{S}$-pseudo-intent $Q \subsetneq P_i$ it is true that $Q'' \subseteq P_i$. To this end let $Q \subsetneq P_i$ be an $\mathcal{S}$-pseudo-intent of $\mathbb{K}$. Then $Q \precnsim P_i \preceq P$, and by definition of P, it is true that $Q = P_j$ for some $j \in \{0, \ldots, i-1\}$. Therefore, $(Q \to Q'') \in \mathcal{L}_i$. Since $\mathcal{L}_i(P_i) = P_i$ and $Q \subseteq P_i$, it must therefore be true that $Q'' \subseteq P_i$.

We have thus shown that P_i is the lectically next $\mathcal{S}$-pseudo-intent after P_{i-1}. Therefore,

$$P_0, \ldots, P_{i+1}$$

are the lectically first $i+2$ $\mathcal{S}$-pseudo-intents of $\mathbb{K}$ and the claim is shown.

From the above claim we can now infer the validity of the theorem. First of all we observe that $\mathbb{K}$ can only have finitely many $\mathcal{S}$-pseudo-intents, since $\mathbb{K}$ itself is finite. Therefore, if $\mathbb{K}$ has n such $\mathcal{S}$-pseudo-intents, then these must be $P_0, \ldots, P_{n-1}$ and $P_n = \mathbf{nil}$. But then

$$\mathcal{L}_n = \{ P_j \to P_j'' \mid j = 0, \ldots, n-1 \} = \mathrm{Can}(\mathbb{K}, \mathcal{S}).$$

Since P_n is **nil**, the value $\mathcal{L}_n$ is returned from `canonical-base` and thus the claim is shown.□

We have to note, however, that the computation as shown in Algorithm 2 may still take time exponential in the size of the input context $\mathbb{K}$ to compute the next $\mathcal{S}$-pseudo-intent. This is mainly because while computing all $\mathcal{S}$-pseudo-intents of $\mathbb{K}$, we also compute all intents of $\mathbb{K}$ as well. It is rather easy to see that the number of intents of a formal context can be exponential in the number of its pseudo-intents, and therefore our implementation may need an exponential delay between the computation of two successive pseudo-intents.

Indeed, it is not known if this exponential delay can be avoided altogether. There is a different approach of computing the canonical base [75], which however also computes all intents of $\mathbb{K}$ during the run. There are also some complexity results with respect to computing the canonical base: enumerating pseudo-intents in lectic order is coNP-hard [40], so an algorithm that computes the canonical base in some sort of lectic order cannot avoid exponential delays, unless P = NP. See also [33] for a practical scenario where, in a certain sense, this phenomenon can be observed. Finally, we note that already recognizing pseudo-intents of a formal context is coNP-hard [19].

On the other hand, it is possible in polynomial time to decide if a base of a formal context is indeed the canonical base. For this, one can use a minimization procedure as discussed in [83], which transforms every base into its corresponding canonical base. The result of this minimization agrees with the original base if and only if the original base is its corresponding canonical base. As this reduction can be done in quadratic time in the size of the original base, this yields a polynomial time method to test if a base is the canonical base.

All in all, it is not clear yet whether computing the canonical base is difficult or not, and it remains an open research problem.

2.5. Attribute Exploration

While we have considered valid implications of our example context $\mathbb{K}_{\mathsf{TNG}}$ we have encountered the implication

$$\{ \mathsf{Human} \} \to \{ \mathsf{Honorable} \}.$$

While neither exact definitions for “human” nor for “honorable” are given, this implication is rather doubtful (even in the context of Star Trek). But still, this implication is a valid implication of $\mathbb{K}_{\mathsf{TNG}}$. On the other hand, one could discuss that this context is not “complete”

in the sense that it lacks relevant counterexamples, and certainly one would find such a counterexample for the above implication in the Star Trek series.

To remedy this incompleteness of $\mathbb{K}_{\mathsf{TNG}}$ one could go and add all characters which have ever occurred in Star Trek *The Next Generation* to $\mathbb{K}_{\mathsf{TNG}}$. While this is doable, it is certainly not practical, especially since it may not be necessary to include *all* characters into $\mathbb{K}_{\mathsf{TNG}}$ to invalidate certain implications.

The illustrated problem may very well occur in practical situations, where one is interested in the valid implications of a certain *domain* which is represented by a collection of *individuals* (Star Trek characters in the above example) with certain *attributes* (those listed in $\mathbb{K}_{\mathsf{TNG}}$). This collection of individuals cannot be listed completely, or at least it is not feasible to do so. However, one can search for single individuals with certain properties.

Let us make this more clear: we assume that our domain is representable by a formal context, which we shall call the *background context* $\mathbb{K}_{\mathsf{back}} = (G_{\mathsf{back}}, M, I_{\mathsf{back}})$. The objects of this formal context are the individuals of our domain of interest. The attributes in the background context are the attributes of the individuals we are *interested in*, in the sense that we want to find implications between those attributes which are valid in the domain. Finally, the incidence of the background context encodes if an individual has a certain attribute in our domain or not.

The task is now to find the implicational theory of the background context. Although we cannot access this context completely, we certainly need a way to access this data. For this, we shall make the following assumption: we are given an *expert* which can *answer questions* of the following type:

> Does the implication $A \to B$ hold in the background context?

The expert may then either agree to this questions, or, if not, she[2] has to provide a counterexample, i. e. a new object g such that $g \in A'$ but $g \notin B'$. Note that this is much more realistic than enumerating all objects of the background context, as we only have to find one object that invalidates the given implication, if it does not hold in the background context.

Within the setting just described, the *attribute exploration* algorithm from formal concept analysis can help [46, 48]. For this algorithm we assume that we have given a *working context* $\mathbb{K} = (G, M, I)$ which is a subcontext of $\mathbb{K}_{\mathsf{back}}$, i. e. $G \subseteq G_{\mathsf{back}}$ and $I = I_{\mathsf{back}} \cap G \times M$.[3] Moreover, we also assume that we have given a set $\mathcal{S} \subseteq \mathrm{Imp}(M)$ of *known implications* which are valid in $\mathbb{K}_{\mathsf{back}}$.

The task now, namely to compute a base of $\mathbb{K}_{\mathsf{back}}$ with background knowledge $\mathcal{S}$, is achieved by the attribute exploration algorithm as follows: the algorithm successively computes implications $(A \to B) \in \mathrm{Imp}(M)$ such that

i. $A \to B$ is valid in $\mathbb{K}$, and

[2] We assume experts to be female unless known otherwise.

[3] Actually, the names of the objects are not relevant here. Indeed, it is sufficient if we can *rename* the objects of $\mathbb{K}$ in a reversible manner such that these conditions hold.

ii. $A \to B$ is not entailed by $\mathcal{S}$.

Intuitively, those implications are *undecided*: they are not invalidated by the objects in $\mathbb{K}$, i. e. by the objects we already know from our domain of interest. On the other hand, $A \to B$ also cannot be inferred from the implications we already know, i. e. from $\mathcal{S}$.

Therefore, we have to ask the expert about the implication $A \to B$. If she decides to accept this implication, $A \to B$ is added to our set of known implications. If she decides to reject the implication, and to provide a counterexample for it, we add it to the formal context. Then, a new implication $A \to B$ is computed which is valid in the working context but does not follow from the known implications.

This algorithm proceeds until no more implications $A \to B$ can be computed. If this is the case, then all valid implications in the working context already follow from the known implications. But this means that the known implications are a base of the background context with background knowledge $\mathcal{S}$: if $X \to Y$ is a valid implication in the background context, then $X \to Y$ is also valid in the working context, and thus it is entailed by the known implications.

Of course, the description of attribute exploration as given above lacks a crucial detail, namely how to compute the implications $A \to B$. Indeed, it turns out that this can be done in a way very similar to how we compute the canonical base of formal context. We shall not discuss this in detail here, as we are going to look into attribute exploration in much more detail in Sections 6 and 7. Instead, we just show an example implementation of attribute exploration in Algorithm 3. There we use the notation $\mathbb{K} + (g, g')$ to denote the formal context which arises from $\mathbb{K} = (G, M, I)$ by adding the object g with attributes g', assuming that $g \notin G$. In other words,

$$\mathbb{K} + (g, g') := (G \cup \{ g \}, M, I \cup \{ (g, m) \mid m \in g' \}).$$

Note that attribute exploration makes use of the Next-Closure algorithm, so we have to provide a linear order $\leqslant_M$ on the attribute set M for a call to `explore-attributes`.

The correctness of the attribute exploration algorithm is stated in the following theorem. However, this theorem states even more, namely a certain kind of *optimality*: the resulting set of implications is not only any base of the background context, but indeed the canonical base of the background context with background knowledge $\mathcal{S}$. Thus, the number of implications *confirmed* by the expert is as small as possible.

2.5.1 Theorem *Let $\mathbb{K} = (G, M, I)$ be a formal context, $\leqslant_M$ a linear order on M, and let $\mathcal{S} \subseteq \mathrm{Imp}(M)$ be such that the expert confirms all implications in $\mathcal{S}$. Then* `explore-attributes` *called with arguments $\mathbb{K}$, $\leqslant_M$ and $\mathcal{S}$ terminates in a finite number of steps. If $\mathbb{K}_{\mathsf{back}}$ denotes the background context of the exploration, then*

$$\texttt{canonical-base}(\mathbb{K}, \leqslant_M, \mathcal{S}) = \mathrm{Can}(\mathbb{K}_{\mathsf{back}}, \mathcal{S}).$$

Algorithm 3 (Attribute Exploration with Background Knowledge)

```
0  define explore-attributes(𝕂 = (G, M, I), ⩽_M, 𝒮 ⊆ Imp(M))
1    ;; conducts attribute exploration with working context 𝕂 and
2    ;; background knowledge 𝒮.
3
4    i := 0,
5    𝕂_i := 𝕂
6    P_i := first-closed-non-intent(𝕂_i, ⩽_M, 𝒮),
7    ℒ_i := ∅
8
9    while P_i ≠ nil do
10
11     if expert confirms P_i → P_i'' then
12       ℒ_{i+1} := ℒ_i ∪ { P_i → P_i'' }
13       𝕂_{i+1} := 𝕂_i
14     else
15       ℒ_{i+1} := ℒ_i
16       𝕂_{i+1} := 𝕂_i + (g, g') ;; g counterexample provided by the expert
17     end
18
19     if P_i ≠ P_i'' then ;; derivation in 𝕂_{i+1}
20       P_{i+1} := P_i
21     else
22       P_{i+1} := next-closed-non-intent(𝕂_{i+1}, ⩽_M, P_i, ℒ_{i+1} ∪ 𝒮)
23     end
24     i := i + 1
25
26   end
27
28   return ℒ_i
29 end
```

We shall not prove this theorem here, but instead refer the reader to standard literature for a proof [41, 45, 48, 92]. Moreover, we shall discuss in Section 6 generalizations of this result, and their proofs then also apply to this theorem.

Note that `explore-attributes` is a generalization of `canonical-base`, and as such inherits all disadvantageous properties of it. In particular, because enumerating pseudo-intents in lectic order cannot be done in polynomial time unless P = NP, it may happen that the time between two questions to the expert may grow exponentially.

CHAPTER 3

Description Logics

In this section, we shall introduce some basic notions of description logics [12] as they are needed for the purpose of this work. This shall include a discussion of some of the basic *syntax* and *semantics* of description logics (Section 3.1), as well as a discussion of *knowledge bases* and *standard reasoning tasks* (Section 3.2). This presentation mostly follows the one from [41].

Our focus in this work will be upon the light-weight description logic $\mathcal{EL}^\bot$, which shall be the target logic for the knowledge we want to extract from data. However, at a later point in this work we shall see that the expressivity of $\mathcal{EL}^\bot$ is not sufficient for our needs, and that we need to consider an extension of $\mathcal{EL}^\bot$ instead. This extension shall be the logic $\mathcal{EL}^\bot_{\text{gfp}}$, an extension of $\mathcal{EL}^\bot$ by *greatest fixpoint semantics*. We shall therefore also include an introduction to the syntax and semantics of $\mathcal{EL}^\bot_{\text{gfp}}$ in this chapter. This will be done in Section 3.3.

3.1. Syntax and Semantics of the Description Logic $\mathcal{EL}^\bot$

We start our introduction to description logics by defining the syntax of the description logic $\mathcal{EL}^\bot$, one of the simplest and least-expressive description logics under consideration. To this end, we choose two disjoint sets N_C and N_R, called the sets of *concept names* and *role names*, respectively. Based upon this choice, we introduce $\mathcal{EL}^\bot$ *concept descriptions* over N_C and N_R as follows.

3.1.1 Definition ($\mathcal{EL}^\bot$ Concept Description) Let N_C and N_R be two disjoint sets. An $\mathcal{EL}$ *concept description* C (over N_C and N_R) is of the form

- $C = A$ for some $A \in N_C$, or
- $C = C_1 \sqcap C_2$ for C_1, C_2 two $\mathcal{EL}$ concept descriptions, or
- $C = \exists r.D$ for an $\mathcal{EL}$ concept description D and $r \in N_R$, or

- $C = \top$.

The constructors are called *conjunction* ($\sqcap$), *existential restriction* ($\exists$) and the *top concept* ($\top$), respectively.

An $\mathcal{EL}^\bot$ concept description C (over N_C and N_R) is then either an $\mathcal{EL}$ concept description or C is of the form $C = \bot$. ♢

We shall occasionally denote the set of all $\mathcal{EL}$ concept descriptions over N_C and N_R by $\mathcal{EL}(N_C, N_R)$, and the set of all $\mathcal{EL}^\bot$ concept descriptions over N_C and N_R by $\mathcal{EL}^\bot(N_C, N_R)$. The same is true for other description logics which we shall introduce at a later point. We shall also often talk just about "concept descriptions" without explicitly mentioning the description logic we are using. If this is the case, then any description logic introduced in this work can be used there.

The sets N_C and N_R of concept and role names are also called the *signature* of the logic. In the following we shall always assume that those sets are finite. Sometimes this signature is extended by another set N_I called the set of *individual names*. Such individual names allow a logic to directly refer to individual elements of a domain. However, this expressiveness is not available in the description logics $\mathcal{EL}$ and $\mathcal{EL}^\bot$, but we shall nevertheless include them in our further discussion as they are needed to define the semantics of assertional axioms. See also Section 3.2.

3.1.2 Example Let $N_C := \{ \mathsf{Cat}, \mathsf{Mouse} \}$ and let $N_R := \{ \mathsf{hunts} \}$. Then examples of $\mathcal{EL}^\bot$ concept descriptions would be

$$\mathsf{Cat},\ \exists \mathsf{hunts}.\mathsf{Mouse},\ \mathsf{Cat} \sqcap \mathsf{Mouse},\ \bot.$$

One can think of these concept descriptions as "describing" things in a certain domain (we shall make this much more precise shortly). For example, the concept description Cat describes everything which is a cat. The concept description $\exists \mathsf{hunts}.\mathsf{Mouse}$ describes everything which hunts a mouse (alternatively: a thing that hunts something which is a mouse). The concept description $\mathsf{Cat} \sqcap \mathsf{Mouse}$ describes things that are both a cat and a mouse. Finally, the concept description $\bot$ just describes nothing. ♢

The intuition of a concept description to "describe things" conveyed in the previous example is very vague, and we shall make it more precise by introducing the notion of an *interpretation*.

3.1.3 Definition (Interpretation) Let N_C and N_R be two disjoint sets. Let N_I be another set, disjoint to both N_C and N_R, called the set of *individual names*. An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ (over N_C, N_R and N_I) consists of a nonempty set $\Delta^\mathcal{I}$ of *elements*, and an *interpretation function* $\cdot^\mathcal{I}$ such that

i. $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ for all $A \in N_C$,

ii. $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$ for all $r \in N_R$, and

iii. $a^\mathcal{I} \in \Delta^\mathcal{I}$ for all $a \in N_I$.

We furthermore make the *unique name assumption*: if $a, b \in N_I$ and $a \neq b$, then $a^\mathcal{I} \neq b^\mathcal{I}$.

We say that $\mathcal{I}$ is finite if $\Delta^\mathcal{I}$ is finite. ♢

If the set N_I in the definition of an interpretation is not important for the current consideration, then we shall set $N_I = \varnothing$ and just speak of an interpretation over N_C and N_R.

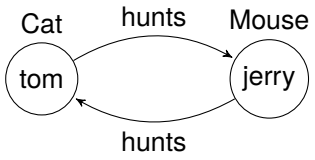
3.1.4 Example Let us consider an example interpretation $\mathcal{I}_{\mathsf{MGM}}$ for our signature $N_C = \{ \mathsf{Cat}, \mathsf{Mouse} \}$ and $N_R = \{ \mathsf{hunts} \}$. For this we set

$$\mathcal{I}_{\mathsf{MGM}} := (\{ \mathsf{tom}, \mathsf{jerry} \}, \cdot^{\mathcal{I}_{\mathsf{MGM}}})$$

where

$$\begin{aligned} \mathsf{Cat}^{\mathcal{I}_{\mathsf{MGM}}} &= \{ \mathsf{tom} \}, \\ \mathsf{Mouse}^{\mathcal{I}_{\mathsf{MGM}}} &= \{ \mathsf{jerry} \}, \\ \mathsf{hunts}^{\mathcal{I}_{\mathsf{MGM}}} &= \{ (\mathsf{tom}, \mathsf{jerry}), (\mathsf{jerry}, \mathsf{tom}) \}. \end{aligned}$$

The interpretation $\mathcal{I}_{\mathsf{MGM}}$ can naturally be represented as a graph, where we consider the elements of $\Delta^{\mathcal{I}_{\mathsf{MGM}}}$ as vertices and the pairs in $\mathsf{hunts}^{\mathcal{I}_{\mathsf{MGM}}}$ as directed edges:



♢

The interpretation function of a given interpretation can now be naturally extended to the set of $\mathcal{EL}$ and $\mathcal{EL}^\bot$ concept descriptions.

3.1.5 Definition (Extensions) Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be an interpretation. Then the *extension* of $\cdot^\mathcal{I}$ to all $\mathcal{EL}^\bot$ concept descriptions is inductively defined as follows: let $C_1, C_2, C \in \mathcal{EL}^\bot(N_C, N_R)$ and $r \in N_R$. Then

i. $\top^\mathcal{I} = \Delta^\mathcal{I}$,

ii. $\bot^\mathcal{I} = \varnothing$,

iii. $(C_1 \sqcap C_2)^\mathcal{I} := C_1^\mathcal{I} \cap C_2^\mathcal{I}$, and

iv. $(\exists r.C)^\mathcal{I} := \{ x \in \Delta^\mathcal{I} \mid \exists y \in \Delta^\mathcal{I} \colon (x, y) \in r^\mathcal{I} \wedge y \in C^\mathcal{I} \}$.

We shall call the set $C^{\mathcal{I}}$ the *extension* of C in $\mathcal{I}$. ♢

The notion of an extension of an $\mathcal{EL}^{\bot}$ concept description in an interpretation now formalizes our previous intuitive understanding that concept descriptions "describe things:" an element $x \in \Delta^{\mathcal{I}}$ is described by an $\mathcal{EL}^{\bot}$ concept description C in $\mathcal{I}$ if and only if $x \in C^{\mathcal{I}}$.

3.1.6 Example Let us consider the example interpretation $\mathcal{I}_{\mathsf{MGM}}$ from Example 3.1.4 again, and let us compute for the example $\mathcal{EL}^{\bot}$ concept descriptions from Example 3.1.2 their extensions. We obtain

$$\begin{aligned}(\exists \mathsf{hunts.Mouse})^{\mathcal{I}_{\mathsf{MGM}}} &= \{\, x \in \Delta^{\mathcal{I}_{\mathsf{MGM}}} \mid \exists y \in \Delta^{\mathcal{I}_{\mathsf{MGM}}} \colon (x, y) \in \mathsf{hunts}^{\mathcal{I}_{\mathsf{MGM}}} \wedge y \in \mathsf{Mouse}^{\mathcal{I}_{\mathsf{MGM}}} \,\} \\ &= \{\, \mathsf{tom} \,\} = \mathsf{Cat}^{\mathcal{I}_{\mathsf{MGM}}}, \\ \mathsf{Cat} \sqcap \mathsf{Mouse}^{\mathcal{I}_{\mathsf{MGM}}} &= \varnothing = \bot^{\mathcal{I}_{\mathsf{MGM}}}.\end{aligned}$$ ♢

For some $\mathcal{EL}^{\bot}$ concept descriptions C, D it may be the case that *for all* interpretations $\mathcal{I}$ it is true that

$$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}.$$

In this case, we say that C is *subsumed by* D, and write $C \sqsubseteq D$. If C is subsumed by D and, in addition, D is subsumed by C, then for all interpretations $\mathcal{I}$ it is true that $C^{\mathcal{I}} = D^{\mathcal{I}}$. In this case we say that C and D are *equivalent*. In this case, we shall write $C \equiv D$.

The definitions we have given so far only apply to the description logics $\mathcal{EL}$ and $\mathcal{EL}^{\bot}$, which are of course not the only ones. Another notable description logic is $\mathcal{ALC}$, a description logic that provides for conjunction, disjunction, negation, $\top$, $\bot$ as well as existential and value restrictions. The definition of the syntax of $\mathcal{ALC}$ is analogous to the one for $\mathcal{EL}$. The semantics of $\mathcal{ALC}$ is again based on interpretations, and the corresponding extension of the interpretation function to all $\mathcal{ALC}$ concept descriptions is given in Table 3.1.

$\mathcal{ALC}$ usually serves as a touchstone for the expressivity of a description logic: a description logic is usually called *inexpressive* if it does not provide (directly or indirectly) all constructors of $\mathcal{ALC}$. A description logic which provides all constructors of $\mathcal{ALC}$ is usually called *expressive*.

Additionally, $\mathcal{ALC}$ played a crucial role in the development of description logics by uncovering a close connection to *modal logics* [17, 26]: in [85] it was shown that $\mathcal{ALC}$ can be considered as a syntactic variant of the multimodal logic $\mathsf{K}_{(\mathsf{m})}$.

3.2. Knowledge Bases and Reasoning

Having defined a description logic, one can use it to state *axioms*. These axioms can be of different nature: they can either state facts about individuals, or they can state facts about concept descriptions in general. In the former case, axioms are called *assertional*

Constructor Name	Syntax	Semantics
top concept	$\top$	$\Delta^{\mathcal{I}}$
bottom concept	$\bot$	$\varnothing$
conjunction	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
disjunction	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\{ x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}} \colon (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}} \}$
value restriction	$\forall r.C$	$\{ x \in \Delta^{\mathcal{I}} \mid \forall y \in \Delta^{\mathcal{I}} \colon (x,y) \in r^{\mathcal{I}} \implies y \in C^{\mathcal{I}} \}$
negation	$\neg C$	$\Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$

Table 3.1.: Syntax Constructors of $\mathcal{ALC}$

axioms, in the latter case they are called *terminological axioms*. A *knowledge base* (or *ontology*) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ then consists of those axioms, where the assertional axioms are collected into an *ABox* $\mathcal{A}$, and where the terminological axioms are collected into a *TBox* $\mathcal{T}$.

If then one has given such a knowledge base $\mathcal{K}$, one can conduct *reasoning* with it. Essentially, reasoning is the extraction of knowledge which is *entailed* by the knowledge base $\mathcal{K}$. Of course, for this to make sense we need to define a semantics for the knowledge base $\mathcal{K}$. Standard reasoning tasks are then *subsumption*, *instance checking*, *consistency checking* and *satisfiability*.

Let us start by formally introducing the notions of *assertional axioms* and *ABoxes*.

3.2.1 Definition (Assertional Axiom, ABox) Let N_C, N_R and N_I be three pairwise disjoint sets. An *assertional axiom* (over N_C, N_R and N_I) is an expression of the form

$$C(a) \quad \text{or} \quad r(a,b)$$

where $a, b \in N_I, r \in N_R$ and $C \in \mathcal{EL}^{\bot}(N_C, N_R)$ is an $\mathcal{EL}^{\bot}$ concept description. An assertional axiom $C(a)$ and $r(a,b)$ *holds* in an interpretation $\mathcal{I}$, written as $\mathcal{I} \models C(a)$ and $\mathcal{I} \models r(a,b)$, if and only if

$$a^{\mathcal{I}} \in C^{\mathcal{I}} \quad \text{and} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}},$$

respectively. An *ABox* $\mathcal{A}$ is then just a set of assertional axioms (over N_C, N_R and N_I). An interpretation $\mathcal{I}$ is a *model* of an ABox $\mathcal{A}$, written $\mathcal{I} \models \mathcal{A}$ if and only if every assertional axiom in $\mathcal{A}$ holds in $\mathcal{I}$. ♢

3.2.2 Example Let us consider Example 3.1.2 again, and let us consider some assertional axioms over N_C, N_R and N_I. For example we can state that tom is a Cat, jerry is a Mouse, that tom hunts jerry and that jerry hunts tom. This can be achieved (in that order) with the following ABox:

$$\mathcal{A}_{\mathsf{MGM}} = \{ \mathsf{Cat}(\mathsf{tom}), \mathsf{Mouse}(\mathsf{jerry}), \mathsf{hunts}(\mathsf{tom}, \mathsf{jerry}), \mathsf{hunts}(\mathsf{jerry}, \mathsf{tom}) \}.$$

The interpretation $\mathcal{I}_{\mathsf{MGM}}$ from Example 3.1.4 is a model of $\mathcal{A}_{\mathsf{MGM}}$, i. e. $\mathcal{I}_{\mathsf{MGM}} \models \mathcal{A}_{\mathsf{MGM}}$. $\diamondsuit$

Complementary to assertional axioms are *terminological axioms*, which state connections between concept descriptions. As already stated, they are collected into *TBoxes*. However, in contrast to ABoxes, there are a lot of different kinds of TBoxes. We shall restrict our attention to two of these, namely *cyclic* TBoxes and *general* TBoxes. For the latter, we shall also introduce the notion of *general concept inclusions*, which is crucial for our further considerations.

3.2.3 Definition (Concept Definitions, Cyclic TBoxes) Let N_C and N_R be two disjoint sets, and let N_D be another set disjoint to both N_C and N_R. A *concept definition* (over N_C, N_R and N_D) is an expression of the form

$$B \equiv D$$

where $B \in N_D$ and D is a concept description over $N_C \cup N_D$ and N_R. The element B is called the *left-hand side* and the concept description D is called the *right-hand side* of the concept definition, respectively.

A *cyclic TBox* $\mathcal{T}$ (over N_C, N_R and N_D) is a set consisting of concept definitions over N_C, N_R and N_D which additionally satisfies the condition that for every $B \in N_D$ there exists exactly one concept definition in $\mathcal{T}$ with B on the left-hand side. The set N_D is called the set of *defined concept names* of $\mathcal{T}$, and is also denoted by $N_D(\mathcal{T})$.

A concept definition $B \equiv D$ is said to *hold* in an interpretation $\mathcal{I}$ over $N_C \cup N_D$ and N_R, written $\mathcal{I} \models (B \equiv D)$, if and only if

$$B^{\mathcal{I}} = D^{\mathcal{I}}$$

is true. The interpretation $\mathcal{I}$ is a *model* for a cyclic TBox $\mathcal{T}$, written $\mathcal{I} \models \mathcal{T}$, if and only if every concept definition in $\mathcal{T}$ holds in $\mathcal{I}$. $\diamondsuit$

A concept definition $B \equiv D$ is true in some interpretation $\mathcal{I}$ if and only if $B^{\mathcal{I}} = D^{\mathcal{I}}$. However, in some cases it may not be possible to exactly define a concept name B in terms of another concept description D. In such cases one can make use of *general concept inclusions*: instead of requiring $B^{\mathcal{I}} = D^{\mathcal{I}}$ one can equivalently state that

$$B^{\mathcal{I}} \subseteq D^{\mathcal{I}} \quad \text{and} \quad B^{\mathcal{I}} \supseteq D^{\mathcal{I}}.$$

If B cannot be defined exactly, at least one of these inclusions may be true. The way to express this is to use general concept inclusion.

3.2.4 Definition (General Concept Inclusion, General TBox) Let N_C and N_R be two disjoint sets. A *general concept inclusion* (GCI) $C \sqsubseteq D$ (over N_C and N_R) consists of two concept

descriptions C, D over N_C and N_R. A GCI $C \sqsubseteq D$ is said to *hold* in an interpretation $\mathcal{I}$ if and only if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.[1]

A collection $\mathcal{T}$ of general concept inclusions is called a *general TBox*. An interpretation $\mathcal{I}$ is said to be a *model* of the general TBox $\mathcal{T}$, written $\mathcal{I} \models \mathcal{T}$, if and only if every GCI $(C \sqsubseteq D) \in \mathcal{T}$ holds in $\mathcal{I}$. In this case we shall also say that $C \sqsubseteq D$ is *valid* in $\mathcal{I}$, or *holds* in $\mathcal{I}$. ♢

In the sense discussed above, concept definitions can be expressed in terms of general concept inclusions. Because of this representation, we can regard cyclic TBoxes as special cases of general TBoxes. Therefore, if we talk about TBoxes in the following we mean either of these notions.

Knowledge bases are now just pairs consisting of an ABox and a TBox.

3.2.5 Definition (Knowledge Base) Let N_C, N_R and N_I be pairwise disjoint sets. Let $\mathcal{A}$ be an ABox over N_C, N_R and N_I, and let $\mathcal{T}$ be a TBox over N_C and N_R. Then the pair

$$\mathcal{K} = (\mathcal{T}, \mathcal{A})$$

is called a *knowledge base* (or *ontology*) (over N_C and N_R). An interpretation is a *model* of $\mathcal{K}$ if and only if it is a model of both $\mathcal{T}$ and $\mathcal{A}$. ♢

Knowledge bases are the core method of description logics to represent knowledge. In the following, we shall consider some of the standard reasoning tasks which can be conducted as soon as a knowledge base is available. As already mentioned, *reasoning* can be seen as the process to extract knowledge from a knowledge base which may be represented only implicitly. The following definitions shall make clear what we mean when we talk about implicitly represented knowledge.

3.2.6 Definition (Consistency Checking) Let $\mathcal{K}$ be a knowledge base. The *consistency checking problem* for $\mathcal{K}$ is to decide whether $\mathcal{K}$ has a model. ♢

3.2.7 Definition (Satisfiability Checking) Let $\mathcal{K}$ be a knowledge base, and let C be an concept description. The *satisfiability checking problem* for $\mathcal{K}$ and C is to decide whether C is satisfiable with respect to $\mathcal{K}$, i. e. whether there exists a model $\mathcal{I}$ of $\mathcal{K}$ such that $C^{\mathcal{I}} \neq \varnothing$.♢

Both checking for consistency and checking satisfiability of certain concept descriptions may be helpful to detect errors in the knowledge base.

3.2.8 Definition (Subsumption Checking) Let $\mathcal{K}$ be a knowledge base, and let C, D be two concept descriptions. Then the *subsumption checking problem* for $\mathcal{K}, C, D$ is to decide whether C is subsumed by D with respect to $\mathcal{K}$, written $C \sqsubseteq_{\mathcal{K}} D$, which means to decide whether $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ is true for all models $\mathcal{I}$ of $\mathcal{K}$. ♢

[1] Notice that there is a possibility for confusion here: recall that we write $C \sqsubseteq D$ if C is subsumed by D. So the fact that $C \sqsubseteq D$ could potentially be confused with the general concept inclusion $C \sqsubseteq D$. However, the former is a statement, while the latter is an expression, so confusing those two is rather unlikely.

Checking subsumption between concept descriptions helps to extract dependencies between concept descriptions in all models of the knowledge base, even if those are not represented explicitly. A related reasoning task is *classification*.

3.2.9 Definition (Classification) Let $\mathcal{K}$ be a knowledge base over N_C and N_R. Then to *classify* $\mathcal{K}$ means to compute the set of all general concept inclusions $A \sqsubseteq B$ such that $A \sqsubseteq_{\mathcal{K}} B$, where $A, B \in N_C$. ♢

Classification and subsumption checking only treat the TBox of the knowledge base. A reasoning task which also involves the ABox is *instance checking*, which may be utilized to find errors in the knowledge base that are related to individuals.

3.2.10 Definition (Instance Checking) Let $\mathcal{K}$ be a knowledge base, let C be a concept description, and let a be an individual name. Then the *instance checking problem* for $\mathcal{K}, C, a$ is to decide if a is an instance of C with respect to $\mathcal{K}$, i. e. whether in all models $\mathcal{I}$ of $\mathcal{K}$ it is true that $a^{\mathcal{I}} \in C^{\mathcal{I}}$. ♢

The complexity of deciding the above decision problems of consistency, satisfiability, subsumption and instance checking is usually used to measure the *reasoning complexity* of a description logic. For expressive description logics it is easy to see that all problems can be reduced to *instance checking*. To this end, let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a knowledge base. Then

i. $\mathcal{K}$ is consistent if and only if a is not an instance of $\bot$ for an arbitrarily chosen individual name a;

ii. C is satisfiable with respect to $\mathcal{K}$ if and only if $(\mathcal{T}, \mathcal{A} \cup \{ C(a) \})$ is consistent, for some individual name a which does not appear in $\mathcal{A}$;

iii. C is subsumed by D with respect to $\mathcal{K}$ if and only if $C \sqcap \neg D$ is not satisfiable with respect to $\mathcal{K}$.

For logics like $\mathcal{ALC}$ it is therefore sufficient to know the complexity of the instance checking problem to know the complexity of the other problems. The instance problem is ExpTime-complete for $\mathcal{ALC}$ [12], and if the TBox of the knowledge base is empty, the complexity of instance checking is PSpace-complete.

For $\mathcal{EL}^{\bot}$ the above reductions do not work anymore, as negation is not available there, and the complexity of the above reasoning problems has more or less to be established separately. It has been shown in [9, 10, 36] that all the above mentioned reasoning problems are tractable, i. e. in PTime.

It is worth noting that $\mathcal{ALC}$, as a variant of the multimodal logic $\mathsf{K}_{(\mathsf{m})}$, has the so-called *finite model property*: if $\mathcal{K}, C$ is an instance of the satisfiability checking problem, then C is satisfiable with respect to $\mathcal{K}$ if and only if there exists a *finite* model $\mathcal{I}$ of $\mathcal{K}$ such that $C^{\mathcal{I}} \neq \varnothing$. This also implies that the subsumption problem can be decided over finite models

only: if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for all finite models $\mathcal{I}$ of $\mathcal{K}$, then $C \sqcap \neg D$ is not satisfiable over some finite model of $\mathcal{K}$. Because of the finite model property of $\mathcal{ALC}$, this implies that $C \sqcap \neg D$ is not satisfiable over any model of $\mathcal{K}$, and thus C is subsumed by D with respect to $\mathcal{K}$. Observe that since $\mathcal{EL}^{\bot}$ is a sub-logic of $\mathcal{ALC}$, it also has the finite model property.

Apart from the standard reasoning tasks, there are many other reasoning problems connected to knowledge bases, like *axiom pinpointing* [74] and *modularization* [50]. A task that is interesting also for our considerations is the computation of *least common subsumers* [16, 100].

3.2.11 Definition (Least Common Subsumer) Let $C_1, \ldots, C_n$ be concept descriptions, and let D be another concept description such that

i. $C_i \sqsubseteq D$ for $i = 1, \ldots, n$ and

ii. for every concept description E such that $C_i \sqsubseteq E$ is true for all $i = 1, \ldots, n$, it is also true that $D \sqsubseteq E$.

Then D is called the *least common subsumer* of $C_1, \ldots, C_n$. ♢

If least common subsumers exist they are unique up to equivalence: if D_1 and D_2 are least common subsumers of $C_1, \ldots, C_n$, then by the second condition of the Definition 3.2.11 it is true that $D_1 \sqsubseteq D_2$ and $D_2 \sqsubseteq D_1$, i. e. $D_1 \equiv D_2$. We shall denote the least common subsumer of $C_1, \ldots, C_n$ by $\mathrm{lcs}(\{ C_1, \ldots, C_n \})$.

Least common subsumers always exist in $\mathcal{EL}$ and $\mathcal{EL}^{\bot}$, i. e. if all concept descriptions mentioned in Definition 3.2.11 are either $\mathcal{EL}$ concept descriptions or $\mathcal{EL}^{\bot}$ concept descriptions, then the least common subsumer always exists [16].

3.3. The Description Logic $\mathcal{EL}^{\bot}_{\mathbf{gfp}}$

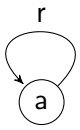
For our considerations on extracting general concept inclusions from finite interpretations we shall soon see that the expressivity of $\mathcal{EL}^{\bot}$ is not sufficient anymore. In particular, we shall see that in $\mathcal{EL}^{\bot}$ *model-based most-specific concept descriptions* do not necessarily exist. These are concept descriptions C which are *most specific* in describing a certain set of individuals X in an interpretation $\mathcal{I}$, i. e. it is true that

i. $X \subseteq C^{\mathcal{I}}$ and

ii. for all concept descriptions D such that $X \subseteq D^{\mathcal{I}}$ it is true that $C \sqsubseteq D$.

We shall discuss model-based most-specific concept descriptions in much more detail in Section 4.2. Here we just show by means of an example that they do not necessarily need to exist in $\mathcal{EL}^{\bot}$.

3.3.1 Example Let us consider $N_C = \varnothing$, $N_R = \{ \mathsf{r} \}$ and the interpretation $\mathcal{I} = (\{ \mathsf{a} \}, \cdot^{\mathcal{I}})$ over N_C and N_R, which is given by $r^{\mathcal{I}} = \{ (\mathsf{a}, \mathsf{a}) \}$. When depicted as a graph, this interpretation is just a single node graph with a loop:



Let $X = \{ \mathsf{a} \}$. Then we can find that *all* $\mathcal{EL}$ concept descriptions which can be formed over the vocabulary N_C and N_R describe X. More specifically, all $\mathcal{EL}$ concept descriptions which can be formed over N_C and N_R are of the form

$$\exists \mathsf{r}^n.\top := \underbrace{\exists \mathsf{r}.\exists \mathsf{r}.\ldots.\exists \mathsf{r}}_{n \text{ times}}.\top$$

for $n \in \mathbb{N}_0$. But then for $m < n$ it is true that

$$\exists \mathsf{r}^n.\top \sqsubseteq \exists \mathsf{r}^m.\top,$$
$$\exists \mathsf{r}^m.\top \not\sqsubseteq \exists \mathsf{r}^n.\top.$$

Because of this, a most-specific concept description for X does not exist. ♢

To remedy this deficit we shall consider an extension of $\mathcal{EL}^{\bot}$ which does guarantee the existence of model-based most-specific concept descriptions, namely the logic $\mathcal{EL}^{\bot}_{\text{gfp}}$ which extends $\mathcal{EL}^{\bot}$ by *cyclic concept descriptions* and *greatest fixpoint semantics* [9, 73].

We shall start by introducing *greatest fixpoint models* for cyclic TBoxes. When we introduced cyclic TBoxes $\mathcal{T}$ in Definition 3.2.3, we have defined that an interpretation $\mathcal{I}$ is a model of $\mathcal{T}$ if and only if every concept definition in $\mathcal{T}$ holds in $\mathcal{I}$. This form of semantics for cyclic TBoxes is called *descriptive* semantics. However, it is possible to further restrict the notion of a model of $\mathcal{T}$, and such a restriction is to use *greatest fixpoint models* instead. To introduce this semantics we need some auxiliary definitions first.

3.3.2 Definition (Primitive Interpretation, Extensions) Let $\mathcal{T}$ be a cyclic TBox over N_C, N_R and N_D. We say that an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a *primitive interpretation* for $\mathcal{T}$ if it is an interpretation over N_C and N_R, i. e. $\cdot^{\mathcal{I}}$ assigns values to the concept names from N_C but not to the defined concept names in N_D. Another interpretation $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ over $N_C \cup N_D$ and N_R is said to *extend* $\mathcal{I}$ if and only if $\mathcal{I}$ and $\mathcal{J}$ have the same set of elements and coincide on N_C and N_R, i. e.

i. $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$,

ii. $A^{\mathcal{I}} = A^{\mathcal{J}}$ is true for all $A \in N_C$, and

iii. $r^{\mathcal{I}} = r^{\mathcal{J}}$ is true for all $r \in N_R$. ♢

Based on this definition we can now give a characterization of models of cyclic TBoxes in terms of fixpoints of certain functions.

Let $\mathcal{T}$ be a cyclic TBox over N_C, N_R and N_D, and let $\mathcal{I}$ be a primitive interpretation. Let us denote with $\mathrm{Ext}(\mathcal{I})$ all interpretations over $N_C \cup N_D$ and N_R that extend $\mathcal{I}$. Then we can naturally define an order relation $\preceq$ on $\mathrm{Ext}(\mathcal{I})$ as follows: let $\mathcal{J}_1, \mathcal{J}_2 \in \mathrm{Ext}(\mathcal{I})$. We define

$$\mathcal{J}_1 \preceq \mathcal{J}_2 \ :\Longleftrightarrow\ A^{\mathcal{J}_1} \subseteq A^{\mathcal{J}_2} \text{ for all } A \in N_D.$$

It can be seen quite easily that with this order relation the ordered set $(\mathrm{Ext}(\mathcal{I}), \preceq)$ becomes a complete lattice: if $\{\, \mathcal{J}_i \mid i \in I \,\} \subseteq \mathrm{Ext}(\mathcal{I})$, then $\sup \{\, \mathcal{J}_i \mid i \in I \,\} = \mathcal{J} \in \mathrm{Ext}(\mathcal{I})$, where

$$A^{\mathcal{J}} := \bigcup_{i \in I} A^{\mathcal{J}_i}$$

for all $A \in N_D$. On this complete lattice we can now define an order-preserving map $f_{\mathcal{I}} \colon \mathrm{Ext}(\mathcal{I}) \to \mathrm{Ext}(\mathcal{I})$ by means of the TBox $\mathcal{T}$. Let $\mathcal{J} \in \mathrm{Ext}(\mathcal{I})$. We define $f_{\mathcal{I}}(\mathcal{J}) = (\Delta^{\mathcal{I}}, \cdot^{f_{\mathcal{I}}(\mathcal{J})}) \in \mathrm{Ext}(\mathcal{I})$ by

$$A^{f_{\mathcal{I}}(\mathcal{J})} := C^{\mathcal{J}}$$

for all $(A \equiv C) \in \mathcal{T}$. Note that since there exists for each $A \in N_D$ exactly one concept definition $A \equiv C$ in $\mathcal{T}$, the function $f_{\mathcal{I}}$ is well-defined.

Recall that $\mathcal{J} \in \mathrm{Ext}(\mathcal{I})$ is a model of $\mathcal{T}$ if and only if

$$A^{\mathcal{J}} = C^{\mathcal{J}}$$

holds for all $(A \equiv C) \in \mathcal{T}$. We can rephrase this condition in terms of *fixpoints* of $f_{\mathcal{I}}$: since $A^{f_{\mathcal{I}}(\mathcal{J})} = C^{\mathcal{J}}$, the interpretation $\mathcal{J}$ is a model of $\mathcal{T}$ if and only if $\mathcal{J}$ is a fixpoint of $f_{\mathcal{I}}$, i. e. $f(\mathcal{J}) = \mathcal{J}$.

As these fixpoints are elements of $\mathrm{Ext}(\mathcal{I})$, they are ordered by $\preceq$. Therefore, intuitively, *greatest fixpoint models* of $\mathcal{T}$ are just greatest fixpoints of $f_{\mathcal{I}}$ with respect to $\preceq$. To ensure their existence, we need to invoke the fixpoint theorem by Tarski [98].

3.3.3 Theorem *Let $(L, \leqslant)$ be a complete lattice and let $f \colon L \to L$ be an order-preserving map, i. e.*

$$x \leqslant y \implies f(x) \leqslant f(y)$$

is true for all $x, y \in L$. Then the fixpoints form a complete sublattice of $(L, \leqslant)$, i. e. the ordered set $(\{\, x \in L \mid f(x) = x \,\}, \leqslant)$ is a complete lattice. In particular, $\{\, x \in L \mid f(x) = x \,\}$ is not empty and has greatest and smallest elements with respect to $\leqslant$.

3.3.4 Definition (Greatest Fixpoint Models) Let $\mathcal{T}$ be a cyclic TBox over N_C, N_R and N_D. Let $\mathcal{I}$ be a primitive interpretation of $\mathcal{T}$. An interpretation $\mathcal{J}$ over $N_C \cup N_D$ and N_R is the *greatest fixpoint model* of $\mathcal{T}$ *extending* $\mathcal{I}$ if and only if $\mathcal{J}$ is the greatest fixpoint of the mapping $f_{\mathcal{I}}$ defined as before. An interpretation $\mathcal{J}$ over $N_C \cup N_D$ and N_R is a *greatest fixpoint model* of $\mathcal{T}$ if it is a greatest fixpoint model of $\mathcal{T}$ extending some primitive interpretation $\mathcal{I}$ of $\mathcal{T}$. ♢

We continue our introduction of $\mathcal{EL}^{\bot}_{\text{gfp}}$ by discussing its semantics. For this we introduce the notion of a *normalized* cyclic TBox.

3.3.5 Definition (Normalized Cyclic TBox) Let $\mathcal{T}$ be a cyclic TBox over N_C, N_R and N_D. Then $\mathcal{T}$ is called *normalized* if and only if for every concept definition $(A \equiv C) \in \mathcal{T}$, the concept description C is of the form

$$C = P_1 \sqcap \cdots \sqcap P_n \sqcap \exists r_1.A_1 \sqcap \cdots \sqcap \exists r_m.A_m$$

for $n, m \in \mathbb{N}_0$, $P_1, \ldots, P_n \in N_C$, $r_1, \ldots, r_m \in N_R$ and $A_1, \ldots, A_m \in N_D$. ♢

We finally have all notions available to define the syntax and semantics of $\mathcal{EL}^{\bot}_{\text{gfp}}$.

3.3.6 Definition ($\mathcal{EL}^{\bot}_{\text{gfp}}$ Concept Descriptions, $\mathcal{EL}^{\bot}_{\text{gfp}}$ Semantics) Let N_C and N_R be two disjoint sets. An $\mathcal{EL}_{gfp}$ *concept description* is an expression of the form $C = (A, \mathcal{T})$, where $\mathcal{T}$ is a normalized cyclic TBox over N_C, N_R and N_D, for some set N_D disjoint to both N_C and N_R, $\mathcal{T}$ only contains $\mathcal{EL}$ concept descriptions, and $A \in N_D$. An $\mathcal{EL}^{\bot}_{gfp}$ *concept description* is either of the form $\bot$ or is an $\mathcal{EL}_{\text{gfp}}$ concept description.

Let $\mathcal{I}$ be an interpretation over N_C and N_R, and let $C = (A, \mathcal{T})$ be an $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept description. Then

$$C^{\mathcal{I}} := A^{\mathcal{J}},$$

where $\mathcal{J}$ is the greatest fixpoint model of $\mathcal{T}$ extending $\mathcal{I}$. ♢

The crucial feature of $\mathcal{EL}^{\bot}_{\text{gfp}}$ is that model-based most-specific concept descriptions always exist. We shall discuss this in more detail in Section 4.2. What we shall do now is to show how the model-based most-specific concept description looks like in Example 3.3.1.

3.3.7 Example Consider the interpretation $\mathcal{I}$ from Example 3.3.1 again. For the set $X = \{ \mathsf{a} \}$ we have found that the concept descriptions

$$C_n := \exists r^n.\top$$

for $n \in \mathbb{N}_0$ satisfy $C_n^{\mathcal{I}} = X$, and that therefore the model-based most-specific concept description for X does not exist in $\mathcal{EL}^{\bot}$. However, we can find a model-based most-specific concept description for X in $\mathcal{EL}^{\bot}_{\text{gfp}}$, namely

$$C := (A, \{ A \equiv \exists r.A \}).$$

Intuitively, one can think of C is an "infinite chain" of existential restrictions, and the possibility to have cyclic concept descriptions in $\mathcal{EL}^{\bot}_{\text{gfp}}$ allows us to express this infinite chain with a finite expression.

The greatest fixpoint model $\mathcal{J}$ for the TBox $\{ A \equiv \exists r.A \}$ extending $\mathcal{I}$ is just given by

$$A^{\mathcal{J}} = \Delta^{\mathcal{I}} = \{ \mathsf{a} \}$$

and therefore $\mathsf{a} \in C^{\mathcal{I}}$. It can also be shown that $C \sqsubseteq C_n$ is true for all $n \in \mathbb{N}_0$, and that all $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions D satisfying $D^{\mathcal{I}} = X$ are equivalent to either C or some C_n. Therefore, C is a model-based most-specific concept description for X. ♢

It may not be apparent in how far the logic $\mathcal{EL}_{\text{gfp}}$ is an extension of $\mathcal{EL}$. To see this, let us define for an $\mathcal{EL}$ concept description C the $\mathcal{EL}_{\text{gfp}}$ concept description

$$C_{\mathcal{EL}_{\text{gfp}}} := (A_C, \{ A_C \equiv C \}).$$

Then $C^{\mathcal{I}} = C^{\mathcal{I}}_{\mathcal{EL}_{\text{gfp}}}$ is true for every interpretation $\mathcal{I}$. Moreover, let us define for two $\mathcal{EL}_{\text{gfp}}$ concept descriptions $D_1 = (A_1, \mathcal{T}_1)$, $D_2 = (A_2, \mathcal{T}_2)$ and $r \in N_R$

$$\begin{aligned} D_1 \sqcap D_2 &:= (A_{D_1 \sqcap D_2}, \mathcal{T}_1 \cup \mathcal{T}_2 \cup \{ A_{D_1 \sqcap D_2} \equiv A_1 \sqcap A_2 \}) \\ \exists r.D_1 &:= (A_{\exists r.D_1}, \mathcal{T}_1 \cup \{ A_{\exists r.D_1} \equiv \exists r.D_1 \}). \end{aligned}$$

Then indeed

$$\begin{aligned} (D_1 \sqcap D_2)^{\mathcal{I}} &= D_1^{\mathcal{I}} \cap D_2^{\mathcal{I}} \\ (\exists r.D_1)^{\mathcal{I}} &= \{ x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}} \colon (x,y) \in r^{\mathcal{I}} \wedge y \in D_1^{\mathcal{I}} \}. \end{aligned}$$

Therefore, the mapping $C \mapsto C_{\mathcal{EL}_{\text{gfp}}}$ maps $\mathcal{EL}(N_C, N_R)$ into a subset of $\mathcal{EL}_{\text{gfp}}(N_C, N_R)$ that behaves like $\mathcal{EL}$ with respect to conjunction and existential restriction, i. e. for every $C, D \in \mathcal{EL}(N_C, N_R)$ and $r \in N_R$ it is true that

$$\begin{aligned} (C \sqcap D)_{\mathcal{EL}_{\text{gfp}}} &\equiv C_{\mathcal{EL}_{\text{gfp}}} \sqcap D_{\mathcal{EL}_{\text{gfp}}}, \\ (\exists r.C)_{\mathcal{EL}_{\text{gfp}}} &\equiv \exists r.C_{\mathcal{EL}_{\text{gfp}}}. \end{aligned}$$

We can therefore regard $\mathcal{EL}_{\text{gfp}}$ as an extension of $\mathcal{EL}$. This of course also means that $\mathcal{EL}^{\bot}_{\text{gfp}}$ can be seen as an extension of $\mathcal{EL}^{\bot}$.

A noteworthy property of $\mathcal{EL}^{\bot}_{\text{gfp}}$ is that least common subsumers always exist, and that they can be computed effectively [7, 8]. In other words, if $C_1, \ldots, C_n$ are $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions, then there exists an $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept description $\operatorname{lcs}(\{ C_1, \ldots, C_n \})$ that is the least common subsumer of $C_1, \ldots, C_n$ in $\mathcal{EL}^{\bot}_{\text{gfp}}$. The existence of $\operatorname{lcs}(\{ C_1, \ldots, C_n \})$ can be shown in a constructive way, giving rise to an effective method to compute the least common subsumer. The construction involves $\mathcal{EL}$ *description graphs* and *products* of these graphs. We shall not go into details here, and refer the interested reader to the literature.

3.4. Unravelling $\mathcal{EL}^{\bot}_{\text{gfp}}$ Concept Descriptions

$\mathcal{EL}^{\bot}_{\text{gfp}}$ has a disadvantage over $\mathcal{EL}^{\bot}$ which impairs its practical use: due to the cyclic nature of $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions their meaning is often hard to grasp, and, particularly, for non-experts in logics, $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions are mostly incomprehensible. Despite that, we need to consider $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions to guarantee the existence of model-based most-specific concept descriptions. Therefore, we cannot dispense with $\mathcal{EL}^{\bot}_{\text{gfp}}$ completely.

Instead, we discuss a method which allows us to transform $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions into $\mathcal{EL}^{\bot}$ concept descriptions in a way suitable for our considerations. This transformation is based on *unravelling* $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions *up to a certain depth*. Concept descriptions obtained by this will always be $\mathcal{EL}^{\bot}$ concept descriptions, which are usually much easier to read and understand.

Recall that $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions C are either of the form $C = \bot$ or $C = (A, \mathcal{T})$ for some cyclic TBox $\mathcal{T}$ and $A \in N_D(\mathcal{T})$. Intuitively, unravelling $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions up to a certain depth means that we "unfold" a potentially cyclic $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept description into $\mathcal{EL}^{\bot}$ concept descriptions with a certain quantifier depth. Of course, $\bot$ is already an $\mathcal{EL}^{\bot}$ concept description, so it suffices to discuss unravelling of $\mathcal{EL}_{\text{gfp}}$ concept descriptions only.

Let us consider an example before we introduce the method in detail.

3.4.1 Example Recall the example concept description from Example 3.3.7

$$C := (A, \{\, A \equiv \exists r.A \,\}).$$

We had argued intuitively that this concept description could be viewed as some kind of "infinite" $\mathcal{EL}^{\bot}$ concept description of the form

$$C \equiv \exists r.\exists r.\exists r. \ldots .$$

Let $d \in \mathbb{N}$. Then an unravelling of C up to depth d would yield a concept description C_d that stems from C by "cutting" the infinite chain of quantifiers after d steps, i. e.

$$C_d = \underbrace{\exists r.\exists r. \ldots . \exists r.}_{d \text{ times}} \top.$$

The role-depth of C_d is then d, because the largest chain of nested existential quantifiers in C_d has depth d. ♢

Let us make the previous argumentation formally precise. To this end, we shall start by introducing the notion of *role depth* of $\mathcal{EL}^{\bot}$ concept descriptions.

3.4.2 Definition (Role Depth of $\mathcal{EL}^{\bot}$ Concept Descriptions) Let N_C and N_R be two disjoint sets. Then the *role-depth* $d(C)$ of a concept description $C \in \mathcal{EL}^{\bot}(N_C, N_R)$ is inductively defined as follows.

i. $d(\bot) = 0$ and $d(A) = 0$ for $A \in N_C$.

ii. $d(C \sqcap D) = \max\{\, d(C), d(D) \,\}$ for $C, D \in \mathcal{EL}^{\bot}(N_C, N_R)$,

iii. $d(\exists r.C) = 1 + d(C)$ for $C \in \mathcal{EL}^{\bot}(N_C, N_R)$. ♢

The general argumentation for formalizing the notion of unravelling up to a certain depth will be achieved in two steps. Firstly, we shall convert a given $\mathcal{EL}_{\text{gfp}}$ concept description C into its $\mathcal{EL}$ *description graph* [9], a notion which we have already encountered before and which we shall now discuss in detail. Those graphs will then be unraveled into *rooted trees*, which in general are infinite. To then obtain the unravelling of C up to a certain depth $d \in \mathbb{N}$, we shall consider only the part of this tree that can be reached from the root in d steps. This subgraph of the $\mathcal{EL}$ description graph of C then gives rise to a concept description C_d, the unravelling of C up to depth d.

3.4.3 Definition ($\mathcal{EL}$ Description Graph) Let N_C and N_R be two disjoint sets, and let $C = (A, \mathcal{T})$ be an $\mathcal{EL}_{\text{gfp}}$ concept description. Then the $\mathcal{EL}$ *description graph* $G_C = (V, E, L)$ of C is defined as follows: recall that for every concept definition $(B \equiv D) \in \mathcal{T}$, the concept description D has the form

$$D = P_1 \sqcap \cdots \sqcap P_n \sqcap \exists r_1.B_1 \sqcap \exists r_m.B_m$$

for $m, n \in \mathbb{N}_0, P_1, \ldots, P_n \in N_C, r_1, \ldots, r_m \in N_R$ and $B_1, \ldots, B_m \in N_D(\mathcal{T})$. We then define

$$\begin{aligned} \text{names}(B) &:= \{\, P_1, \ldots, P_n \,\}, \\ \text{succ}_r(B) &:= \{\, B_i \mid 1 \leqslant i \leqslant m, r_i = r \,\} \end{aligned}$$

for every $r \in N_R$. To define $G_C = (V, E, L)$ we set

i. $V := N_D(\mathcal{T})$,

ii. $E := \{\, (B_1, r, B_2) \mid B_1, B_2 \in N_D(\mathcal{T}), B_2 \in \text{succ}_r(B_1) \,\}$,

iii. $L(B) := \text{names}(B)$ for each $B \in N_D(\mathcal{T})$.

We shall call V the *vertices*, E the *edges* and L the *labeling function* of the description graph G_C. ♢

It is easy to see that every $\mathcal{EL}$ description graph can be converted into a corresponding concept description by just reverting the process described in the above definition. Moreover, if a concept description is converted into an $\mathcal{EL}$ description graph, and then back into a concept description, then the resulting concept description is equivalent to the original one.

To now define the notion of an unravelling of a given $\mathcal{EL}$ description graph $G_C = (V, E, L)$ we shall follow the definitions as in [41] and introduce the notion of a *directed path*

in G_C. Such a directed path is a word $w = A_1 r_1 A_2 r_2 \dots r_n A_{n+1}$ for some $n \in \mathbb{N}_0$ such that $A_1, \dots, A_{n+1} \in V$ and for all $1 \leqslant i \leqslant n$ it is true that $(A_i, r_i, A_{i+1}) \in E$. We shall then say that the directed path w *starts* at A_1 and *ends* at A_{n+1}, and shall also write $\delta(w) := A_{n+1}$ and call $\delta(w)$ the *destination* of w. The *length* $\mathrm{len}(w)$ of w is defined to be n.

3.4.4 Definition (Unravelling of $\mathcal{EL}$ Description Graphs and of $\mathcal{EL}$ Concept Descriptions) Let N_C and N_R be two disjoint sets, let $C \in \mathcal{EL}_{\mathrm{gfp}}(N_C, N_R)$, and let $G_C = (V, E, L)$ the $\mathcal{EL}$ description graph of C. Then the *unravelling* of G_C is defined to be the graph $G_\infty = (V_\infty, E_\infty, L_\infty)$ where

i. V_∞ is the set of all directed paths in G_C,

ii. $E_\infty := \{ (w, r, wrB) \mid w, wrB \in V_\infty \}$,

iii. $L_\infty(w) := L(\delta(w))$ for $w \in V_\infty$.

Let $d \in \mathbb{N}_0$. Then the *unravelling* of G_C up to depth d is defined as $G_d := (V_d, E_d, L_d)$ where

i. $V_d := \{ w \in V_\infty \mid \mathrm{len}(w) \leqslant d \}$,

ii. $E_d := \{ (A, r, B) \in E_\infty \mid A, B \in V_d \}$,

iii. $L_d(w) := L_\infty(w)$ for $w \in V_d$.

Finally, the *unravelling* of C up to depth d is defined as the $\mathcal{EL}$ concept description C_d that corresponds to the unravelling of G_C up to depth d. In addition, we set $\bot_d := \bot$ for each $d \in \mathbb{N}_0$. ♢

A crucial result about unravellings C_d of concept descriptions C is the following lemma.

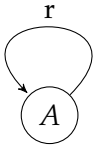
3.4.5 Lemma (Lemma 5.3 of [41]) *Let C be an $\mathcal{EL}^{\bot}_{gfp}$ concept description. Then for all $d \in \mathbb{N}_0$ it is true that $C \sqsubseteq C_d$.*

To finish this chapter, let us consider an example which illustrates the process of unravelling.

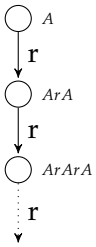
3.4.6 Example Let us revisit Example 3.4.1 again and let us compute unravellings of the $\mathcal{EL}$ concept description $C = (A, \{ A \equiv \exists r.A \})$ formally now. Its $\mathcal{EL}$ description graph is then

$$G_C = (\{ A \}, \{ (A, r, A) \}, A \mapsto \varnothing),$$

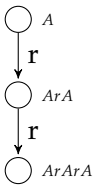
and can be depicted as follows



The unravelling of G_C is then the graph G_∞ which corresponds to the following picture



where the path to which every node corresponds is depicted right of the node. Then, for $d = 3$, the unravelling of G_C up to depth 3 would just be the graph



and the corresponding $\mathcal{EL}^{\perp}$ concept description is

$$C_3 = \exists r.\exists r.\exists r.\top.$$ ♢

CHAPTER 4

Axiomatizing Valid General Concept Inclusions of Finite Interpretations

Our considerations about extracting general concept inclusions from erroneous data will be based on previous results obtained by Baader and Distel [41] on extracting all *valid* general concept inclusions from a given finite interpretation. In this section, we shall therefore review the notions and results from this work that are necessary for our own.

The problem of extracting all valid general concept inclusions from a finite interpretation can be made more precise as follows. Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation over N_C and N_R, i. e. $\Delta^{\mathcal{I}}$ is a finite set. The task then is to find the set of all general concept inclusions $C \sqsubseteq D$ with $C, D \in \mathcal{EL}^{\bot}(N_C, N_R)$ which are valid in $\mathcal{I}$.

Of course, the set of all valid general concept inclusions is infinite in general. This is because if $C \sqsubseteq D$ holds in $\mathcal{I}$, and $r \in N_R$, then $\exists r.C \sqsubseteq \exists r.D$ holds in $\mathcal{I}$ as well. Such an infinite set is hardly usable to represent knowledge suitable for machine consumption. Therefore, the considerations in [41] concentrate on finding *finite bases* of $\mathcal{I}$, i. e. sets of valid general concept inclusions of $\mathcal{I}$ that are also *complete*. We shall introduce these notions briefly in Section 4.1.

One of the main results of [41] then is that finite bases for finite interpretations $\mathcal{I}$ always exist, and we shall discuss them in Section 4.3. These results have been obtained by exploiting a close connection between description logics and formal concept analysis. It is therefore crucial that we introduce this connection first, and we shall do so in Section 4.2. In particular, we shall talk about *induced contexts* and *model-based most-specific concept descriptions*.

4.1. Bases of General Concept Inclusions

General concept inclusions have a *model-based semantics*, i. e. their semantics is defined in terms of being valid in some interpretation. We can therefore introduce the notions of *entailment* and *completeness* as follows. Also notice the similarity of this definition to Definition 2.4.1.

4.1.1 Definition Let $\mathcal{L} \cup \{ C \sqsubseteq D \}$ be a set of general concept inclusions over N_C and N_R. We shall say that $\mathcal{L}$ *entails* $C \sqsubseteq D$, written $\mathcal{L} \models (C \sqsubseteq D)$ if and only if for all interpretations over N_C and N_R it is true that if $\mathcal{I} \models \mathcal{L}$, then $\mathcal{I} \models \{ C \sqsubseteq D \}$ as well.

Let $\mathcal{K}$ be another set of general concept inclusions over N_C and N_R. Then $\mathcal{K}$ is said to be *sound* for $\mathcal{L}$ if and only if all general concept inclusions in $\mathcal{K}$ are entailed by $\mathcal{L}$. $\mathcal{K}$ is said to be *complete* for $\mathcal{L}$ if and only if all general concept inclusions in $\mathcal{L}$ are *entailed* by $\mathcal{K}$. $\mathcal{K}$ is said to be a base for $\mathcal{L}$ if and only if $\mathcal{K}$ is sound and complete for $\mathcal{L}$. ♢

Let $\mathcal{I}$ be a finite interpretation over N_C and N_R, and let us denote with $\mathrm{Th}(\mathcal{I})$ the set of all $\mathcal{EL}^{\bot}_{\mathrm{gfp}}$ general concept inclusions over N_C and N_R which are valid in $\mathcal{I}$, i. e.

$$\mathrm{Th}(\mathcal{I}) := \{ C \sqsubseteq D \mid C, D \in \mathcal{EL}^{\bot}_{\mathrm{gfp}}(N_C, N_R), C^{\mathcal{I}} \subseteq D^{\mathcal{I}} \}.$$

Let $\mathcal{K}$ be a set of general concept inclusions over N_C and N_R. If $\mathcal{K}$ is a base of $\mathrm{Th}(\mathcal{I})$, we shall simply say that $\mathcal{K}$ is a *base* of $\mathcal{I}$. If $\mathcal{K}$ consists of $\mathcal{EL}^{\bot}$ general concept inclusions only, we shall say that $\mathcal{K}$ is an $\mathcal{EL}^{\bot}$ *base* of $\mathcal{I}$. Otherwise, we shall occasionally say that $\mathcal{K}$ is an $\mathcal{EL}^{\bot}_{gfp}$ *base* of $\mathcal{I}$.

Notice that in the case that $\mathcal{K}$ is a base of $\mathcal{I}$, all general concept inclusions in $\mathcal{K}$ have to hold in $\mathcal{I}$: the set $\mathrm{Th}(\mathcal{I})$ is *closed under entailment* in the sense that every $\mathcal{EL}^{\bot}_{\mathrm{gfp}}$ general concept inclusion over N_C and N_R which is entailed by $\mathrm{Th}(\mathcal{I})$ is already contained in this set. Therefore, if $\mathcal{K}$ is sound for $\mathrm{Th}(\mathcal{I})$, it must be contained in this set and thus $\mathcal{K}$ is a set of general concept inclusions which are valid in $\mathcal{I}$. Moreover, since $\mathcal{K}$ is complete for $\mathrm{Th}(\mathcal{I})$, every $\mathcal{EL}^{\bot}_{\mathrm{gfp}}$ general concept inclusion over N_C and N_R that holds in $\mathcal{I}$ is entailed by $\mathcal{K}$.

4.2. Linking Formal Concept Analysis and Description Logics

Description logics and formal concept analysis are connected by a number of similar notions. As an example, let us consider a formal context $\mathbb{K} = (G, M, I)$ and a set $A \subseteq M$. The set A' then is the set of all objects of $\mathbb{K}$ which have all the attributes in A. We can view this fact from another perspective: if $A = \{ m_1, \ldots, m_n \}$, then we can think of the attributes $m_1, \ldots, m_n$ as *propositions*, and the fact that $(g, m) \in I$ as saying that g *satisfies* the proposition m. Then $g \in A'$ means that g *satisfies* the conjunction of all propositions in A.

Let us reformulate this using description logics. To this end, let us define $N_C := M$ and $N_R = \varnothing$. Then we can think of $\mathbb{K}$ as an interpretation $\mathcal{I}_{\mathbb{K}} = (G, \cdot^{\mathcal{I}_{\mathbb{K}}})$ where

$$m^{\mathcal{I}_{\mathbb{K}}} := \{ g \in G \mid (g, m) \in I \} = \{ m \}'. \tag{4.1}$$

Then we have $A' = (m_1 \sqcap \cdots \sqcap m_n)^{\mathcal{I}_{\mathbb{K}}}$ for all finite $A = \{ m_1, \ldots, m_n \} \subseteq M$. Indeed, if we would consider a description logic that only allows for conjunction $\sqcap$, then we can view finite formal contexts, derivation of sets of attributes and even implications as special cases of finite interpretations, extensions of concept descriptions and general concept inclusions.

Thus the derivation operator $(\cdot)' \colon \mathfrak{P}(M) \to \mathfrak{P}(G)$ naturally corresponds to computing the extension of concept descriptions in interpretations. However, the other derivation operator $(\cdot)' \colon \mathfrak{P}(G) \to \mathfrak{P}(M)$ does not have such a correspondence in description logics. This gap shall be filled by considering *model-based most-specific concept descriptions*, which we introduce in Section 4.2.1.

The connection between description logics and formal concept analysis expressed in (4.1) only works in one direction: it allows to represent basic notions of formal concept analysis in terms of description logics, but not vice versa. Even if we restrict our attention to the rather light-weight description logic $\mathcal{EL}^{\bot}$, it is not clear how to represent an interpretation by means of notions from formal concept analysis.

To approach this issue, we shall introduce *induced contexts* in Section 4.2.2. Such contexts allow to express tight connections between the notions of formal concept analysis and description logics, and, since induced contexts are just formal contexts, still allow the application of standard methods from formal concept analysis, such as the extraction of bases. This fact will be exploited when we discuss the computation of finite bases in Section 4.3.

4.2.1. Model-Based Most-Specific Concept Descriptions

Let $\mathbb{K} = (G, M, I)$, and let us try to motivate how to find a natural correspondence of the derivation operator $(\cdot)' \colon \mathfrak{P}(G) \to \mathfrak{P}(M)$ within description logics. Let $B \subseteq G$ be a set of objects of $\mathbb{K}$. Then the set $A := B'$ can be thought of as the *most-specific* set of attributes that *describe* B, i. e.

i. $B \subseteq A'$, i. e. A *describes* B, and

ii. for all sets $C \subseteq M$ that satisfy $B \subseteq C'$ (that describe B) it is true that $C \subseteq A$, (A contains *more* attributes than C, i. e. is *more specific*).

The last point is true because if $B \subseteq C'$, then by Lemma 2.1.9 it is true that $C \subseteq B' = A$. Notice that the description of A as a most-specific description of B is also a characterization, i. e. if A is the most-specific description of B in the above sense, then $A = B'$.

To mimic this *most-specific description* in description logics, Baader and Distel introduce the notion of *most-specific concept descriptions*.

4.2.1 Definition (Model-Based Most-Specific Concept Description) Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation, and let $X \subseteq \Delta^{\mathcal{I}}$. A *model-based most-specific concept description* of X in $\mathcal{I}$ is a concept description C such that

i. $X \subseteq C^{\mathcal{I}}$, and

ii. for each concept description D satisfying $X \subseteq D^{\mathcal{I}}$ it is true that $C \sqsubseteq D$, i. e. C is subsumed by D. ♢

If a model-based most-specific concept description C for X in $\mathcal{I}$ exists, it is unique up to equivalence: if D is another such model-based most-specific concept description, then $C \sqsubseteq D$ and $D \sqsubseteq C$, by the last condition of the definition. Therefore, $C \equiv D$. Because of this, we can talk about *the* model-based most-specific concept description of X in $\mathcal{I}$, and shall denote it with $X^{\mathcal{I}}$, to stress the similarity to the derivation operator from formal concept analysis. We shall also write $X^{\mathcal{I}\mathcal{I}}$ instead of $(X^{\mathcal{I}})^{\mathcal{I}}$ and $C^{\mathcal{I}\mathcal{I}}$ instead of $(C^{\mathcal{I}})^{\mathcal{I}}$ for syntactic convenience.

The existence of model-based most-specific concept descriptions, however, is not clear per se, and the choice of the description logic in which we seek for model-based most-specific concept descriptions is crucial here: if we only consider $\mathcal{EL}^{\bot}$ concept descriptions, then model-based most-specific concept descriptions do not necessarily exist, as is shown in Example 3.3.1. However, if we allow all concept descriptions in Definition 4.2.1 to be $\mathcal{EL}_{\text{gfp}}$ or $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions, then the existence of model-based most-specific concept descriptions can be guaranteed.

4.2.2 Theorem (Theorem 4.7 of [41]) *Model-based most-specific concept descriptions exist in $\mathcal{EL}_{gfp}$ and $\mathcal{EL}^{\bot}_{gfp}$ for all finite interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ and sets $X \subseteq \Delta^{\mathcal{I}}$, and they can be computed effectively.*

The computation of model-based most-specific concept descriptions can be achieved using $\mathcal{EL}$ *description graphs*, least common subsumers and *simulations* [9, 41]. See [41, Section 4.1.2] for details on this.

We have motivated model-based most-specific concept descriptions by most-specific descriptions in formal contexts, and for this we have made use of the fact that the derivation operators form a Galois connection. It is therefore only natural to expect that model-based most-specific concept descriptions are part of a Galois connection, too. However, we have to notice that we cannot expect to obtain a Galois connection in the sense of Section 2.2, simply because the relation $\sqsubseteq$ is not antisymmetric, and thus not an order relation: it may be the case that $C \sqsubseteq D$ and $D \sqsubseteq C$, but $D \neq C$. We can remedy this fact by considering concept descriptions only *up to equivalence*: instead of a single concept description C, we always consider the set $[C]$ of all concept descriptions which are equivalent to C. Then $[C] \sqsubseteq [D]$ is well-defined for all concept descriptions C and D, and $\sqsubseteq$ indeed yields an order relation this way. This is only a technical detail, however, and we shall not make it explicit in our following considerations.

4.2.3 Lemma (Lemma 4.1 of [41]) *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation over N_C and N_R, $X \subseteq \Delta^{\mathcal{I}}$ and C an $\mathcal{EL}^{\bot}_{gfp}$ concept description over N_C and N_R. Then*

$$X \subseteq C^{\mathcal{I}} \iff X^{\mathcal{I}} \sqsubseteq C. \tag{4.2}$$

In particular, for $X, Y \subseteq \Delta^{\mathcal{I}}$ and for $\mathcal{EL}^{\bot}_{gfp}$ concept descriptions C, D over N_C and N_R, it is true that

i. $X \subseteq Y \implies X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}$,

ii. $C \sqsubseteq D \implies C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$,

iii. $X \subseteq X^{\mathcal{I}\mathcal{I}}$,

iv. $C^{\mathcal{I}\mathcal{I}} \sqsubseteq C$,

v. $X^{\mathcal{I}} \equiv X^{\mathcal{I}\mathcal{I}\mathcal{I}}$,

vi. $C^{\mathcal{I}} = C^{\mathcal{I}\mathcal{I}\mathcal{I}}$.

Proof We only show (4.2), the other claims follow from Lemma 2.2.3 and the above-made considerations. If $X \subseteq C^{\mathcal{I}}$, then $X^{\mathcal{I}} \sqsubseteq C$ because $X^{\mathcal{I}}$ is by definition the most-specific concept description that contains X in its extension. Conversely, if $X^{\mathcal{I}} \sqsubseteq C$, then by definition $X^{\mathcal{I}\mathcal{I}} \subseteq C^{\mathcal{I}}$. But since $X^{\mathcal{I}}$ is the model-based most-specific concept description of X in $\mathcal{I}$, it contains X in its extension, i. e. $X \subseteq X^{\mathcal{I}\mathcal{I}}$. Therefore, $X \subseteq C^{\mathcal{I}}$. □

Another useful property is the following, rather technical proposition.

4.2.4 Proposition (Lemma 4.2 of [41]) *Let $\mathcal{I}$ be an interpretation over N_C and N_R, and let C, D be $\mathcal{EL}^{\bot}_{gfp}$ concept descriptions over N_C and N_R and let $r \in N_R$. Then*

i. $(C \sqcap D)^{\mathcal{I}} = (C^{\mathcal{I}\mathcal{I}} \sqcap D)^{\mathcal{I}}$, *and*

ii. $(\exists r.C)^{\mathcal{I}} = (\exists r.C^{\mathcal{I}\mathcal{I}})^{\mathcal{I}}$.

Proof For the first claim we use Lemma 4.2.3 and obtain

$$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} = C^{\mathcal{I}\mathcal{I}\mathcal{I}} \cap D^{\mathcal{I}} = (C^{\mathcal{I}\mathcal{I}} \sqcap D)^{\mathcal{I}}.$$

For the second one we observe that

$$\begin{aligned}(\exists r.C^{\mathcal{I}\mathcal{I}})^{\mathcal{I}} &= \{\, x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}} \colon (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}\mathcal{I}\mathcal{I}} \,\} \\ &= \{\, x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}} \colon (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}} \,\} \\ &= (\exists r.C)^{\mathcal{I}},\end{aligned}$$

again because of $C^{\mathcal{I}} = C^{\mathcal{I}\mathcal{I}\mathcal{I}}$ from Lemma 4.2.3. □

4.2.2. Induced Contexts

We already have seen how formal contexts can be represented as interpretations. In this section we shall introduce the approach of Baader and Distel of *induced contexts*, which provides the inverse direction, i. e. which allows to represent interpretations as formal contexts. The notion of induced contexts also was used implicitly by works of Prediger [77] in her study on *terminological attribute logic*.

4.2.5 Definition (Induced Context) Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation over N_C and N_R, and let M be a set of concept descriptions over N_C and N_R. The *induced context* of $\mathcal{I}$ and M is the formal context $\mathbb{K}_{\mathcal{I},M} = (\Delta^{\mathcal{I}}, M, \nabla)$, where for $x \in \Delta^{\mathcal{I}}$ and $C \in M$

$$(x, C) \in \nabla \;:\Longleftrightarrow\; x \in C^{\mathcal{I}}. \qquad \diamondsuit$$

Induced formal contexts do not necessarily represent the interpretation $\mathcal{I}$ completely; indeed, what is represented of $\mathcal{I}$ heavily depends on the choice of the set M of concept descriptions. We later shall see that we can choose this set M to obtain a close connection between bases of $\mathbb{K}_{\mathcal{I},M}$ and bases of $\mathcal{I}$.

We start our considerations about induced contexts by introducing some auxiliary notions first. For a finite set $U \subseteq M$ we define the set

$$\bigsqcap U := \begin{cases} \top & \text{if } U = \emptyset, \\ \bigsqcap_{V \in U} V & \text{otherwise.} \end{cases}$$

We call $\bigsqcap U$ the *concept description defined by* U. Furthermore, for a concept description C we define the *projection* of C onto M as

$$\mathrm{pr}_M(C) := \{\, D \in M \mid C \sqsubseteq D \,\}.$$

Concept descriptions defined by subsets of M together with projections capture some kind of notion of *upper approximation* in terms of M: if C is a concept description, then the most-specific concept description D satisfying $C \sqsubseteq D$ that can be defined by a subset of M is given by

$$D = \bigsqcap \mathrm{pr}_M(C).$$

This looks familiar to our introductory motivation for model-based most-specific concept descriptions, and indeed there are similarities. One of them is that the mappings $U \mapsto \bigsqcap U$ and $C \mapsto \mathrm{pr}_M(C)$ satisfy the main condition of an antitone Galois connection.

4.2.6 Lemma *Let M be a finite set of concept descriptions over N_C and N_R. Then for each $U \subseteq M$ and each concept description C over N_C and N_R it is true that*

$$C \sqsubseteq \bigsqcap U \iff U \subseteq \mathrm{pr}_M(C).$$

In particular, the following statements holds for all $U, V \subseteq M$ and all concept descriptions C, D over N_C and N_R.

i. $C \sqsubseteq D \implies \mathrm{pr}_M(D) \subseteq \mathrm{pr}_M(C)$,

ii. $U \subseteq V \implies \bigsqcap V \sqsubseteq \bigsqcap U$,

iii. $C \sqsubseteq \bigsqcap \mathrm{pr}_M(C)$,

iv. $U \subseteq \mathrm{pr}_M(\bigsqcap U)$.

Proof Assume $C \sqsubseteq \bigsqcap U$. Then $\mathrm{pr}_M(\bigsqcap U) \subseteq \mathrm{pr}_M(C)$, since every concept description $D \in M$ satisfying $\bigsqcap U \sqsubseteq D$ also satisfies $C \sqsubseteq D$. Furthermore, $U \subseteq \mathrm{pr}_M(\bigsqcap U)$, since for each $F \in U$ it is true that $\bigsqcap U \sqsubseteq F$. Thus

$$U \subseteq \mathrm{pr}_M(\bigsqcap U) \subseteq \mathrm{pr}_M(C).$$

For the converse direction, assume that $U \subseteq \mathrm{pr}_M(C)$. Then $\bigsqcap \mathrm{pr}_M(C) \sqsubseteq \bigsqcap U$. Since for each $D \in \mathrm{pr}_M(C)$ it is true that $C \sqsubseteq D$, we also have $C \sqsubseteq \bigsqcap \mathrm{pr}_M(C)$. In sum, we obtain

$$C \sqsubseteq \bigsqcap \mathrm{pr}_M(C) \sqsubseteq \bigsqcap U.$$ □

For certain concept descriptions C, the upper approximation provided by $\bigsqcap \mathrm{pr}_M(C)$ coincides with C. Those concept descriptions are exactly those which are *expressible in terms of* M, i. e. there exists a subset $N \subseteq M$ such that $C \equiv \bigsqcap N$.

4.2.7 Lemma ([41]) *Let $M \cup \{ C \}$ be a set of concept descriptions over N_C and N_R. Then C is expressible in terms of M if and only if*

$$C \equiv \bigsqcap \mathrm{pr}_M(C).$$

Proof Clearly, if $C \equiv \bigsqcap \mathrm{pr}_M(C)$, then C is expressible in terms of M. Conversely, if C is expressible in terms of M, then $C \equiv \bigsqcap N$ for some $N \subseteq M$. Then $C \sqsubseteq D$ for all $D \in N$, and therefore $N \subseteq \mathrm{pr}_M(C)$. By Lemma 4.2.6, it is thus true that

$$C \sqsubseteq \bigsqcap \mathrm{pr}_M(C) \sqsubseteq \bigsqcap N \equiv C$$

and therefore $C \equiv \bigsqcap \mathrm{pr}_M(C)$. □

We can now state some connections between the derivation operators of an induced context on one side, and computing the extension of a concept description as well as model-based most-specific concept descriptions on the other. These results are rather technical but necessary for our further considerations. We include the proofs of these statements here, as they are rather simple and may help to better understand the corresponding claims.

4.2.8 Proposition (Lemma 4.11 and 4.12 of [41]) *Let $\mathcal{I}$ be a finite interpretation and M be a finite set of concept descriptions. Then for every concept description expressible in terms of M it is true that*

$$C^{\mathcal{I}} = (\mathrm{pr}_M(C))',$$

and for $O \subseteq \Delta^{\mathcal{I}}$ it is true that

$$O' = \mathrm{pr}_M(O^{\mathcal{I}}),$$

where the derivation is conducted in $\mathbb{K}_{\mathcal{I},M}$.

Proof Since C is expressible in terms of M, Lemma 4.2.7 yields $C \equiv \bigsqcap \mathrm{pr}_M(C)$. Thus

$$\begin{aligned} x \in C^{\mathcal{I}} &\iff x \in (\textstyle\bigsqcap \mathrm{pr}_M(C))^{\mathcal{I}} \\ &\iff \forall D \in \mathrm{pr}_M(C) \colon x \in D^{\mathcal{I}} \\ &\iff x \in (\mathrm{pr}_M(C))', \end{aligned}$$

since $(\mathrm{pr}_M(C))' = \{\, x \in \Delta^{\mathcal{I}} \mid \forall D \in \mathrm{pr}_M(C) \colon x \in D^{\mathcal{I}} \,\}$.
If $O \subseteq \Delta^{\mathcal{I}}$, then

$$\begin{aligned} D \in O' &\iff \forall g \in O \colon g \in D^{\mathcal{I}} \\ &\iff O \subseteq D^{\mathcal{I}} \\ &\iff O^{\mathcal{I}} \sqsubseteq D \\ &\iff D \in \mathrm{pr}_M(O^{\mathcal{I}}), \end{aligned}$$

where $O \subseteq D^{\mathcal{I}} \iff O^{\mathcal{I}} \sqsubseteq D$ holds due to Lemma 4.2.3. □

4.2.9 Proposition (Lemma 4.10 and 4.11 of [41]) *Let $\mathcal{I}$ be a finite interpretation and let M be a finite set of concept descriptions. Then each $B \subseteq M$ satisfies*

$$B' = (\textstyle\bigsqcap B)^{\mathcal{I}},$$

and if $A \subseteq \Delta^{\mathcal{I}}$ is such that $A^{\mathcal{I}}$ is expressible in terms of M, then

$$\textstyle\bigsqcap A' \equiv A^{\mathcal{I}},$$

where all derivations are conducted in $\mathbb{K}_{\mathcal{I},M} = (\Delta^{\mathcal{I}}, M, \nabla)$.

Proof Observe that $x \in B'$ if and only if $x \in C^{\mathcal{I}}$ for all $C \in B$. Therefore

$$x \in B' \iff \forall C \in B \colon x \in C^{\mathcal{I}} \iff x \in \bigcap_{C \in B} C^{\mathcal{I}} = (\textstyle\bigsqcap B)^{\mathcal{I}},$$

and therefore $B' = (\bigsqcap B)^{\mathcal{I}}$.

If $A \subseteq \Delta^{\mathcal{I}}$ is such that $A^{\mathcal{I}}$ is expressible in terms of M, then by Lemma 4.2.7 it is true that

$$A^{\mathcal{I}} \equiv \textstyle\bigsqcap \mathrm{pr}_M(A^{\mathcal{I}}).$$

By Proposition 4.2.8, $\mathrm{pr}_M(A^{\mathcal{I}}) = A'$, and thus $A^{\mathcal{I}} \equiv \bigsqcap A'$ as required. □

4.2.10 Proposition *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation and let M be a set of concept descriptions. Let $A \subseteq \Delta^{\mathcal{I}}$ such that $A^{\mathcal{I}}$ is expressible in terms of M. Then $A^{\mathcal{I}\mathcal{I}} = A''$, where the derivations are conducted in $\mathbb{K}_{\mathcal{I},M}$.*

Proof Again, by Lemma 4.2.7 we have $A^{\mathcal{I}} \equiv \bigsqcap \mathrm{pr}_M(A^{\mathcal{I}})$ and thus

$$\begin{aligned} A^{\mathcal{I}\mathcal{I}} &= \left(\bigsqcap \mathrm{pr}_M(A^{\mathcal{I}})\right)^{\mathcal{I}} \\ &= \mathrm{pr}_M(A^{\mathcal{I}})' \\ &= A'' \end{aligned}$$

by Proposition 4.2.8 and Proposition 4.2.9. □

We can rephrase some of the above results as follows. Let $\mathcal{I}$ be a finite interpretation and let us call a concept description C a *model-based most-specific concept description* of $\mathcal{I}$ if it is the model-based most-specific concept description of some subset of $\Delta^{\mathcal{I}}$. Note that C is a model-based most-specific concept description of $\mathcal{I}$ if and only if $C \equiv C^{\mathcal{I}\mathcal{I}}$.

Let M be a set of concept descriptions such that all model-based most-specific concept descriptions are expressible in terms of M. If we then identify equivalent model-based most-specific concept descriptions and order them by $\sqsubseteq$, then the resulting ordered set is dually isomorphic to the lattice of intents of $\mathbb{K}_{\mathcal{I},M}$. Note that with $\mathrm{Int}(\mathbb{K}_{\mathcal{I},M})$ we denote the set of intents of $\mathbb{K}_{\mathcal{I},M}$.

4.2.11 Corollary (contains Corollary 4.13 of [41]) *Let $\mathcal{I}$ be a finite interpretation and let M be a set of concept descriptions such that model-based most-specific concept descriptions of $\mathcal{I}$ are expressible in terms of M. Denote with $\mathcal{M}$ the set of all model-based most-specific concept descriptions considered up to equivalence. Then the mapping*

$$\begin{array}{rccc} \varphi\colon & \mathrm{Int}(\mathbb{K}_{\mathcal{I},M}) & \to & \mathcal{M} \\ & U & \mapsto & \bigsqcap U \end{array}$$

is an order-isomorphism between $(\mathrm{Int}(\mathbb{K}_{\mathcal{I},M}), \subseteq)$ and $(\mathcal{M}, \sqsupseteq)$, where

$$\varphi^{-1}(C) = \mathrm{pr}_M(C) \quad (C \in \mathcal{M}).$$

In particular this means

i. $\bigsqcap U \in \mathcal{M}$ *for all* $U \in \mathrm{Int}(\mathbb{K}_{\mathcal{I},M})$,

ii. $\mathrm{pr}_M(C) \in \mathrm{Int}(\mathbb{K}_{\mathcal{I},M})$ *for all* $C \in \mathcal{M}$,

iii. $U \subseteq V$ *implies* $\bigsqcap U \sqsupseteq \bigsqcap V$ *for all* $U, V \subseteq M$,

iv. $C \sqsubseteq D$ *implies* $\mathrm{pr}_M(C) \supseteq \mathrm{pr}_M(D)$ *for all* $C, D \in \mathcal{M}$,

v. $\mathrm{pr}_M(\bigsqcap U) = U$ *for all* $U \in \mathrm{Int}(\mathbb{K}_{\mathcal{I},M})$,

vi. $\bigsqcap \mathrm{pr}_M(C) \equiv C$ *for each* $C \in \mathcal{M}$.

Additionally,

$$\begin{aligned} U'' &= \mathrm{pr}_M((\bigsqcap U)^{\mathcal{I}\mathcal{I}}), \\ C^{\mathcal{I}\mathcal{I}} &= \bigsqcap(\mathrm{pr}_M(C))'' \end{aligned} \tag{4.3}$$

is true for all $U \subseteq M$ *and all concept descriptions* C *expressible in terms of* M*, and where the derivations are done in* $\mathbb{K}_{\mathcal{I},M}$.

Proof Claims (iii) and (iv) are already contained in Lemma 4.2.6, and (vi) is just Lemma 4.2.7 again. We show the other claims step by step.

For (i) let $U \in \mathrm{Int}(\mathbb{K}_{\mathcal{I},M})$. Then $U = U''$, and thus

$$\bigsqcap U = \bigsqcap U'' \equiv (U')^{\mathcal{I}} = (\bigsqcap U)^{\mathcal{I}\mathcal{I}}$$

by Proposition 4.2.9. Thus $U \in \mathcal{M}$ up to equivalence.

For (ii) let $C \in \mathcal{M}$. Then $C \equiv C^{\mathcal{I}\mathcal{I}}$ and C is expressible in terms of M. From Proposition 4.2.8 it follows

$$\begin{aligned} \mathrm{pr}_M(C) &= \mathrm{pr}_M(C^{\mathcal{I}\mathcal{I}}) \\ &= (C^{\mathcal{I}})' \\ &= \mathrm{pr}_M(C)'' \end{aligned}$$

and thus $\mathrm{pr}_M(C) \in \mathrm{Int}(\mathbb{K}_{\mathcal{I},M})$.

For (v) let again $U \in \mathrm{Int}(\mathbb{K}_{\mathcal{I},M})$. We first observe that $U \subseteq \mathrm{pr}_M(\bigsqcap U)$ by Lemma 4.2.6. Furthermore, for each concept description D it is true that

$$\begin{aligned} D \in \mathrm{pr}_M(\bigsqcap U) &\iff \bigsqcap U \sqsubseteq D \\ &\implies (\bigsqcap U)^{\mathcal{I}} \subseteq D^{\mathcal{I}} \\ &\iff U' \subseteq \{ D \}' \\ &\iff U'' \supseteq \{ D \}'' \ni D \\ &\iff D \in U'' = U, \end{aligned}$$

using Proposition 4.2.9 for $(\bigsqcap U)^{\mathcal{I}} = U'$, and the definition of $\mathbb{K}_{\mathcal{I},M}$ to obtain $D^{\mathcal{I}} = \{ D \}'$. Thus, $\mathrm{pr}_M(\bigsqcap U) \subseteq U$ and equality follows.

For the equations given in (4.3) we observe

$$\mathrm{pr}_M((\bigsqcap U)^{\mathcal{I}\mathcal{I}}) = \mathrm{pr}_M((U')^{\mathcal{I}})$$

$$= U''$$

by Proposition 4.2.9 and Proposition 4.2.8, and

$$\begin{aligned}\bigsqcap(\mathrm{pr}_M(C))'' &\equiv (\mathrm{pr}_M(C)')^{\mathcal{I}} \\ &= C^{\mathcal{I}\mathcal{I}},\end{aligned}$$

again because of Proposition 4.2.9 and Proposition 4.2.8, for every $U \subseteq M$ and every concept description C expressible in terms of M. □

The equivalence $\bigsqcap(\mathrm{pr}_M(C))'' \equiv C^{\mathcal{I}\mathcal{I}}$ does not hold in general for concept descriptions C, as the following trivial example shows.

4.2.12 Example Let $N_C = \varnothing$ and $N_R = \{\,\mathsf{r}\,\}$, and let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation over N_C and N_R with $\Delta^{\mathcal{I}} = \{\,x\,\}$ and $r^{\mathcal{I}} = \varnothing$. Then the model-based most-specific concept descriptions of $\mathcal{I}$ are, up to equivalence, just $\top$ and $\bot$. Let $M = \{\,\bot\,\}$. Then clearly all model-based most-specific concept descriptions of $\mathcal{I}$ are expressible in terms of M. Then

$$\mathbb{K}_{\mathcal{I},M} = \begin{array}{c|c} & \bot \\ \hline x & \cdot \end{array}$$

Now consider $C = \exists r.\top$. Then on the one hand,

$$C^{\mathcal{I}\mathcal{I}} = \varnothing^{\mathcal{I}} = \bot,$$

but on the other hand

$$\bigsqcap \mathrm{pr}_M(C)'' = \bigsqcap \varnothing'' = \bigsqcap \varnothing = \top,$$

so $C^{\mathcal{I}\mathcal{I}} \neq \bigsqcap \mathrm{pr}_M(C)''$. ♢

A useful consequence of Corollary 4.2.11 is the following result.

4.2.13 Lemma *Let $\mathcal{I}$ be a finite interpretation, and let $U \subseteq M_{\mathcal{I}}$. Then*

$$(\bigsqcap U)^{\mathcal{I}\mathcal{I}} = \bigsqcap U'',$$

where the derivations are done in $\mathbb{K}_{\mathcal{I}}$.

Proof Clearly $\bigsqcap U$ is expressible in terms of $M_{\mathcal{I}}$. Thus Corollary 4.2.11 yields

$$(\bigsqcap U)^{\mathcal{I}\mathcal{I}} = \bigsqcap(\mathrm{pr}_{M_{\mathcal{I}}}(\bigsqcap U))''.$$

By Proposition 4.2.8 it is true that $\mathrm{pr}_{M_\mathcal{I}}(\bigsqcap U)' = (\bigsqcap U)^\mathcal{I}$, thus

$$\begin{aligned}(\bigsqcap U)^{\mathcal{I}\mathcal{I}} &= \bigsqcap((\bigsqcap U)^\mathcal{I})' \\ &= \bigsqcap U''\end{aligned}$$

where $(\bigsqcap U)^\mathcal{I} = U'$ is true due to Proposition 4.2.9. □

4.3. Computing Bases of Valid GCIs of a Finite Interpretation

Using the notions of model-based most-specific concept descriptions and induced contexts, we are finally prepared to introduce some of the main results of [41] on computing bases of finite interpretations. The main idea behind these results is to use ideas and methods from formal concept analysis, either by simulating them in a description logic setting, or by transforming the initially given interpretation into formal contexts and applying standard methods from formal concept analysis to it.

Recall that for a (finite) formal context $\mathbb{K} = (G, M, I)$ the set

$$\{\, A \to A'' \mid A \subseteq M \,\}$$

is always a base of $\mathbb{K}$. This is because every valid implication $(A \to B) \in \mathrm{Th}(\mathbb{K})$ already follows from $A \to A''$, because if $\mathbb{K} \models (A \to B)$, then $A' \subseteq B'$, i. e. $B \subseteq A''$ and thus $\{\, A \to A'' \,\} \models (A \to B)$. Having introduced model-based most-specific concept descriptions, we are able to simulate this result in terms of description logics as follows.

4.3.1 Lemma (Lemma 4.3 of [41]) *Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be an interpretation, and let $C \sqsubseteq D$ be a general concept inclusion that is valid in $\mathcal{I}$. Then $C \sqsubseteq C^{\mathcal{I}\mathcal{I}}$ is valid in $\mathcal{I}$ as well, and $C \sqsubseteq D$ follows from $C \sqsubseteq C^{\mathcal{I}\mathcal{I}}$.*

The following statement is then a simple corollary.

4.3.2 Corollary *Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be an interpretation over N_C and N_R. Then*

$$\mathcal{B}_0 := \{\, C \sqsubseteq C^{\mathcal{I}\mathcal{I}} \mid C \in \mathcal{EL}^\bot_{gfp}(N_C, N_R), C \neq \bot \,\} \tag{4.4}$$

is a base of $\mathcal{I}$.

Of course, this base is not finite in general, i. e. if $N_R \neq \varnothing$. However, based on this result, Baader and Distel investigate subsets of $\mathcal{B}_0$ and finally arrive at a finite base. The first step into this direction is to show that considering only $\mathcal{EL}^\bot$ concept descriptions is enough, as described in the next theorem. The main advantage of this result is that we can now use induction over the premises of general concept inclusions.

4.3.3 Theorem (Theorem 5.7 of [41]) *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation over N_C and N_R. Then*

$$\mathcal{B}_1 := \{\, C \sqsubseteq C^{\mathcal{I}\mathcal{I}} \mid C \in \mathcal{EL}^{\bot}(N_C, N_R), C \neq \bot \,\} \tag{4.5}$$

is a base of $\mathcal{I}$.

The proof of this theorem is quite involved, and again makes use of $\mathcal{EL}$ description graphs and simulations between them. We shall not go into details here, and refer the reader to [41, Section 5.1.1].

The base $\mathcal{B}_1$ still is not finite in general. To achieve finiteness, we consider a particular finite set $M_{\mathcal{I}}$ of concept descriptions which turns out to be enough, in the sense that we only need to consider general concept inclusions $C \sqsubseteq C^{\mathcal{I}\mathcal{I}}$ where $C = \bigsqcap U$ for some $U \subseteq M_{\mathcal{I}}$. Since $M_{\mathcal{I}}$ is finite, the resulting set of general concept inclusions is finite and therefore yields a finite base of $\mathcal{I}$.

4.3.4 Definition ($M_{\mathcal{I}}$) Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation over N_C and N_R. Then

$$M_{\mathcal{I}} := N_C \cup \{\, \bot \,\} \cup \{\, \exists r.X^{\mathcal{I}} \mid r \in N_R, X \subseteq \Delta^{\mathcal{I}}, X \neq \varnothing \,\}. \qquad \diamondsuit$$

The definition of $M_{\mathcal{I}}$ seems to be incomprehensible at first. However, since this set will play a major role for our further considerations, we shall give some intuition why it is suitable for our purpose the way it is defined.

Note that $M_{\mathcal{I}}$ is finite since $\mathcal{I}$ is finite, and thus there are only finitely many subsets of $\Delta^{\mathcal{I}}$. Furthermore notice that $M_{\mathcal{I}}$ can be computed using the Next-Closure algorithm from Theorem 2.4.11. More precisely, we can compute all concept descriptions $X^{\mathcal{I}}$ by noticing that $X^{\mathcal{I}} \equiv X^{\mathcal{I}\mathcal{I}\mathcal{I}}$, and we can compute the sets $X^{\mathcal{I}\mathcal{I}}$ using Next-Closure because the mapping $X \mapsto X^{\mathcal{I}\mathcal{I}}$ is a closure operator on $\mathfrak{P}(M_{\mathcal{I}})$.

Before we show how the set $M_{\mathcal{I}}$ helps in finding finite bases, we note an important property of it.

4.3.5 Lemma (Lemma 5.9 of [41]) *Let $\mathcal{I}$ be a finite interpretation and let C be a model-based most-specific concept description of $\mathcal{I}$. Then C is expressible in terms of $M_{\mathcal{I}}$.*

Let us define

$$\mathcal{B}_2 := \{\, \bigsqcap U \sqsubseteq (\bigsqcap U)^{\mathcal{I}\mathcal{I}} \mid U \subseteq M_{\mathcal{I}} \,\}. \tag{4.6}$$

Then clearly $\mathcal{B}_2 \models (C \sqsubseteq C^{\mathcal{I}\mathcal{I}})$ for $C \in N_C$ or $C = \bot$. For $C = D \sqcap E$, and assuming by induction that $\mathcal{B}_2 \models (D \sqsubseteq D^{\mathcal{I}\mathcal{I}})$ and $\mathcal{B}_2 \models (E \sqsubseteq E^{\mathcal{I}\mathcal{I}})$, we can find that

$$\mathcal{B}_2 \models (D \sqcap E \sqsubseteq D^{\mathcal{I}\mathcal{I}} \sqcap E^{\mathcal{I}\mathcal{I}}).$$

But then $D^{\mathcal{I}\mathcal{I}} \sqcap E^{\mathcal{I}\mathcal{I}}$ is expressible in terms of $M_{\mathcal{I}}$ (as a conjunction of model-based most-specific concept descriptions, using Lemma 4.3.5), so

$$\mathcal{B}_2 \models ((D^{\mathcal{I}\mathcal{I}} \sqcap E^{\mathcal{I}\mathcal{I}}) \sqsubseteq (D^{\mathcal{I}\mathcal{I}} \sqcap E^{\mathcal{I}\mathcal{I}})^{\mathcal{I}\mathcal{I}}).$$

Using Proposition 4.2.4 we obtain $(D^{\mathcal{I}\mathcal{I}} \sqcap E^{\mathcal{I}\mathcal{I}})^{\mathcal{I}\mathcal{I}} \equiv (D \sqcap E)^{\mathcal{I}\mathcal{I}}$, so all in all

$$\mathcal{B}_2 \models (D \sqcap E \sqsubseteq (D \sqcap E)^{\mathcal{I}\mathcal{I}}).$$

Notice that the main arguments here are Proposition 4.2.4 and that all model-based most-specific concept descriptions are expressible in terms of $M_{\mathcal{I}}$.

If $C = \exists r.D$, and assuming that $\mathcal{B}_2 \models (D \sqsubseteq D^{\mathcal{I}\mathcal{I}})$, we first obtain

$$\mathcal{B}_2 \models (\exists r.C \sqsubseteq \exists r.C^{\mathcal{I}\mathcal{I}}). \tag{4.7}$$

But then $(\exists r.C^{\mathcal{I}\mathcal{I}}) \in M_{\mathcal{I}}$ up to equivalence, so

$$\mathcal{B}_2 \models (\exists r.C^{\mathcal{I}\mathcal{I}} \sqsubseteq (\exists r.C^{\mathcal{I}\mathcal{I}})^{\mathcal{I}\mathcal{I}}).$$

Using Proposition 4.2.4 again we obtain $(\exists r.C^{\mathcal{I}\mathcal{I}})^{\mathcal{I}\mathcal{I}} \equiv (\exists r.C)^{\mathcal{I}\mathcal{I}}$, so

$$\mathcal{B}_2 \models (\exists r.C \sqsubseteq (\exists r.C)^{\mathcal{I}\mathcal{I}}).$$

Notice that the crucial property in that argumentation is that $M_{\mathcal{I}}$ contains concept descriptions of the form $\exists r.C^{\mathcal{I}\mathcal{I}}$, and that Proposition 4.2.4 has been used again.

The preceding argument then shows the following claim.

4.3.6 Theorem (Theorem 5.10 of [41]) *Let $\mathcal{I}$ be a finite interpretation. Then $\mathcal{B}_2$ as defined in Equation* (4.6) *is a finite base of $\mathcal{I}$.*

A practical disadvantage of the finite base $\mathcal{B}_2$ is its size, which may be exponential in $|M_{\mathcal{I}}|$, which itself may be exponential in the size of $\Delta^{\mathcal{I}}$. To remedy this, we use methods from formal concept analysis to extract bases from formal contexts. In particular, recall that the canonical base of a formal context is minimal in size among all bases of a formal context, and that it can be computed effectively. Having this in mind, we further observe that if we consider the induced formal context $\mathbb{K}_{\mathcal{I}} := \mathbb{K}_{\mathcal{I},M_{\mathcal{I}}}$, then the set $\mathcal{L} := \{ A \to A'' \mid A \subseteq M_{\mathcal{I}} \}$ is a base of $\mathbb{K}_{\mathcal{I}}$, and that

$$\mathcal{B}_2 = \bigsqcap \mathcal{L} := \{ \bigsqcap A \sqsubseteq \bigsqcap A'' \mid (A \to A'') \in \mathcal{L} \}.$$

Recall that $\bigsqcap A'' \equiv (\bigsqcap A)^{\mathcal{I}\mathcal{I}}$ by Corollary 4.2.11.

We can generalize this observation as follows: if $\mathcal{L} \subseteq \mathrm{Th}(\mathbb{K}_{\mathcal{I}})$ is a base of $\mathbb{K}_{\mathcal{I}}$ which only contains implications of the form $U \to U''$, then the set $\bigsqcap \mathcal{L}$ defined as

$$\bigsqcap \mathcal{L} := \{ \bigsqcap U \sqsubseteq (\bigsqcap U)^{\mathcal{I}\mathcal{I}} \mid (U \to U'') \in \mathcal{L} \}$$

is a base of $\mathbb{K}_\mathcal{I}$. Note that then $\bigsqcap \mathcal{L}$ is always a subset of $\mathcal{B}_2$, but $\bigsqcap \mathcal{L}$ may be much smaller than $\mathcal{B}_2$, for example if $\mathcal{L}$ is irredundant or even minimal.

However, there is a redundancy in $\mathcal{L}$ which cannot be removed this way: if $C, D \in M_\mathcal{I}$ such that C is subsumed by D, then the implication $\{ C \} \to \{ D \}$ will always be true in $\mathbb{K}_\mathcal{I}$. But this means that this implication has to be contained implicitly or explicitly in any base of $\mathbb{K}_\mathcal{I}$. On the other hand, the resulting GCI $C \sqsubseteq D$ is trivial, and thus dispensable.

We can alleviate this situation by making use of bases with background knowledge. The background knowledge we are interested in would be

$$\mathcal{S}_\mathcal{I} := \{ \{ C \} \to \{ D \} \mid C, D \in M_\mathcal{I}, C \sqsubseteq D \}. \tag{4.8}$$

A base of $\mathbb{K}_\mathcal{I}$ with background knowledge $\mathcal{S}_\mathcal{I}$ now does not have to contain the information about the implications in $\mathcal{S}_\mathcal{I}$ anymore, and may thus may be smaller than a base without this background knowledge.

4.3.7 Theorem (Theorem 5.12 of [41]) *Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be a finite interpretation, and let $\mathcal{L}$ be a base of $\mathbb{K}_\mathcal{I}$ with background knowledge $\mathcal{S}_\mathcal{I}$. Assume that $\mathcal{L}$ only contains implications of the form $U \to U''$ for some $U \subseteq M_\mathcal{I}$. Then $\bigsqcap \mathcal{L}$ is a finite base of $\mathcal{I}$.*

We can extend this connection between bases of $\mathbb{K}_\mathcal{I}$ and bases of $\mathcal{I}$ even more: if $\mathcal{L}$ is the canonical base of $\mathbb{K}_\mathcal{I}$ with background knowledge $\mathcal{S}_\mathcal{I}$, then $\bigsqcap \mathcal{L}$ is a minimal base of $\mathcal{I}$.

4.3.8 Theorem (Theorem 5.18 of [41]) *Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be a finite interpretation, and define*

$$\mathcal{B} := \bigsqcap \{ A \to A'' \mid (A \to A'') \in \mathrm{Can}(\mathbb{K}_\mathcal{I}, \mathcal{S}_\mathcal{I}) \}.$$

Then $\mathcal{B}$ is a minimal base of $\mathcal{I}$.

So far, all bases we have obtained were $\mathcal{EL}^\bot_{\mathrm{gfp}}$-bases, i. e. the GCIs contained in these bases where allowed to contain proper $\mathcal{EL}^\bot_{\mathrm{gfp}}$ concept descriptions. From a logical point of view this is not a problem. However, $\mathcal{EL}^\bot_{\mathrm{gfp}}$ concept descriptions are inherently harder to read, since they allow for "local recursion" within concept descriptions. This may be undesired, as those concept descriptions may have to be inspected by domain experts for their validity, and those experts may not necessarily be experts in logic as well.

On the other hand, $\mathcal{EL}^\bot$ concept descriptions are much easier to read, and thus obtaining $\mathcal{EL}^\bot$ bases instead of $\mathcal{EL}^\bot_{\mathrm{gfp}}$ bases may be much more desirable. For this, Baader and Distel discuss a way to obtain such $\mathcal{EL}^\bot$ bases from arbitrary $\mathcal{EL}^\bot_{\mathrm{gfp}}$ bases by *unravelling*.

The crucial observation towards obtaining $\mathcal{EL}^\bot$ bases from $\mathcal{EL}^\bot_{\mathrm{gfp}}$ bases is that given a finite interpretation $\mathcal{I}$ and a concept description C it is true for $d \in \mathbb{N}_0$ "large enough" that $C^\mathcal{I} = C^\mathcal{I}_d$. Recall that C_d denotes the unravelling of C up to depth d.

4.3.9 Lemma (Lemma 5.5 of [41]) *Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be a finite interpretation, and let $C = (A, \mathcal{T})$ be an $\mathcal{EL}_{gfp}$ concept description. Then for $d = |N_D(\mathcal{T})| \cdot |\Delta^\mathcal{I}| + 1$ it is true that $C^\mathcal{I} = C^\mathcal{I}_d$.*

Secondly, unravelling up to depth d respects the structure of $\mathcal{EL}^\bot$ concept descriptions, as formulated in the following lemma.

4.3.10 Lemma (Lemma 5.19 of [41]) *Let C, D be two $\mathcal{EL}_{gfp}$ concept descriptions. Then*

i. $(\exists r.C)_d \equiv \exists r.C_{d-1}$,

ii. $(C \sqcap D)_d \equiv C_d \sqcap D_d$.

To now unravel an $\mathcal{EL}^\bot_{\text{gfp}}$ base $\mathcal{B}$ of $\mathcal{I}$ the idea is to just unravel every GCI $(C \sqsubseteq D) \in \mathcal{B}$ "deep enough", i. e. replacing these GCIs by $C_d \sqsubseteq D_d$, where d is chosen as in Lemma 4.3.9. This, however, may not be enough, as we may not be able anymore to entail GCIs of the form $(X^\mathcal{I})_d \sqsubseteq X^\mathcal{I}$ for $X \subseteq \Delta^\mathcal{I}$ from the base thus obtained. To remedy this, some extra GCIs need to be added.

4.3.11 Theorem (Theorem 5.21 of [41]) *Let $\mathcal{I}$ be a finite interpretation and let $\mathcal{B}$ be a finite $\mathcal{EL}^\bot_{gfp}$ base of $\mathcal{I}$. Then*

$$\mathcal{B}_\mathsf{u} := \{\, C_d \sqsubseteq (C^{\mathcal{I}\mathcal{I}})_d \mid (C \sqsubseteq D) \in \mathcal{B} \,\} \cup \{\, (X^\mathcal{I})_d \sqsubseteq (X^\mathcal{I})_{d+1} \mid X \subseteq \Delta^\mathcal{I}, X \neq \varnothing \,\}$$

is a finite $\mathcal{EL}^\bot$ base of $\mathcal{I}$, where $d \in \mathbb{N}_0$ is defined as in Lemma 4.3.9.

We shall only give some intuition why this theorem is correct, as we shall discuss its proof when we generalize it to bases of confident GCIs in Section 5.2.6. An important observation is that the set

$$\mathcal{X} := \{\, (X^\mathcal{I})_d \sqsubseteq (X^\mathcal{I})_{d+1} \mid X \subseteq \Delta^\mathcal{I}, X \neq \varnothing \,\}$$

satisfies for all $X \subseteq \Delta^\mathcal{I}$

i. $\mathcal{X} \models ((X^\mathcal{I})_k \sqsubseteq (X^\mathcal{I})_{k+1})$ for all $k \in \mathbb{N}_0, k \geqslant d$, and

ii. $\mathcal{X} \models ((X^\mathcal{I})_d \sqsubseteq X^\mathcal{I})$.

The first property can be shown by induction over k, and for the second property we observe that if $\mathcal{J}$ is a finite interpretation such that $\mathcal{J} \models \mathcal{X}$, then by the first property

$$((X^\mathcal{I})_d)^\mathcal{J} \subseteq ((X^\mathcal{I})_{d+1})^\mathcal{J} \subseteq ((X^\mathcal{I})_{d+2})^\mathcal{J} \subseteq \dots$$

Since $\mathcal{J}$ is finite, for k large enough it is true that $((X^\mathcal{I})_k)^\mathcal{J} = ((X^\mathcal{I})_{k+1})^\mathcal{J}$ and thus

$$((X^\mathcal{I})_k)^\mathcal{J} = (X^\mathcal{I})^\mathcal{J}.$$

Thus, $\mathcal{J} \models ((X^\mathcal{I})_d \sqsubseteq X^\mathcal{I})$ and therefore $\mathcal{X} \models ((X^\mathcal{I})_d \sqsubseteq X^\mathcal{I})$, because $\mathcal{EL}^\bot$ has the finite model property.

But then if $(C \sqsubseteq D) \in \mathcal{B}$, then $\mathcal{B}_\mathsf{u} \models ((C^{\mathcal{I}\mathcal{I}})_d \sqsubseteq C^{\mathcal{I}\mathcal{I}})$ by the argument just shown, and $\mathcal{B}_\mathsf{u} \models (C_d \sqsubseteq (C^{\mathcal{I}\mathcal{I}})_d)$, because this GCI is contained in $\mathcal{B}_\mathsf{u}$. Thus

$$\mathcal{B}_\mathsf{u} \models (C \sqsubseteq C_d \sqsubseteq (C^{\mathcal{I}\mathcal{I}})_d \sqsubseteq C^{\mathcal{I}\mathcal{I}}),$$

and Lemma 4.3.1 yields $\mathcal{B}_\mathsf{u} \models (C \sqsubseteq D)$. Thus, $\mathcal{B}_d$ entails all GCIs in $\mathcal{B}$, and since $\mathcal{B}$ is complete for $\mathcal{I}$, $\mathcal{B}_\mathsf{u}$ is complete for $\mathcal{I}$ as well.

Chapter 5

Axiomatizing General Concept Inclusions with High Confidence

The results obtained by Baader and Distel about computing finite bases of finite interpretations are not only interesting from a theoretical point of view. Although we have skipped most of the details, all the relevant results are *effective* in the sense that the obtained bases can in principle be computed by computers. Thus, these results may also be interesting for practical applications.

A possible application of the results of Baader and Distel is to compute bases from *Linked Open Data* [24], a format for representing data as used by the semantic web [22, 43, 59]. This data format consists of RDF triples, and can thus be thought of as an edge-labeled graph. As such, it is very similar to interpretations, and thus we can use the results by Baader and Distel here.

As a first contribution of this thesis we have implemented the major results on computing finite bases as described previously, and applied them to a particular data set of the *Linked Open Data Cloud*, namely to a subset of the DBpedia data set [25]. In Section 5.1 we describe this experiment in detail and show what Distel's results yield when applied to this data set. This experiment has also been discussed previously [28, 33].

One conclusion from this experiment is that the results by Baader and Distel are very sensitive to *errors* in the data. This is actually not surprising: bases of finite interpretations only contain general concept inclusions which are valid in the data, and if there is as little as a single counterexample to a given general concept inclusion in the data, it will not be contained in any base.

If those counterexamples are erroneous, however, then this can cause problems. Not only that otherwise valid general concept inclusions are not obtained by Distel's approach anymore. Sporadic erroneous counterexamples may also cause GCIs found during the computation of bases to be rather complicated, because those GCIs have to avoid those erroneous counterexamples using complicated concept descriptions.

To remedy or at least to alleviate this effect of erroneous counterexamples we shall

consider an extension of Baader and Distel's results. This extension tries to find bases of GCIs which are not necessarily valid in the given interpretation, but instead enjoy a *high confidence* therein. The notion of *confidence* is borrowed from data-mining [1], more precisely from the theory of *association rules*, and allows to measure how much an association rule is allowed to ignore counterexamples. We shall transfer the notion of confidence to general concept inclusions, and shall then try to find bases for those *GCIs with high confidence*.

For this we shall make use of results obtained by Luxenburger from his work on *partial implications* [68, 69], and extensions thereof [97]. Partial implications can be thought of as implications considered together with their confidence in some particular formal context. We shall discuss in Section 5.2.2 how these results allow us to obtain bases of all implications which have *high confidence* in some given formal context.

In Section 5.2.3 then we show how these ideas can be simulated in $\mathcal{EL}^{\bot}_{\text{gfp}}$ to find bases for GCIs with high confidence. Moreover, we shall show in Section 5.2.4 how bases of implications with high confidence yield bases of GCIs with high confidence. We shall also discuss a way to *complete* sets of GCIs, i. e. how to obtain a set of valid GCIs that makes a given set of GCIs complete. This result is quite similar to Theorem 4.3.7, and we shall discuss it in Section 5.2.5. Finally, we shall see how to obtain $\mathcal{EL}^{\bot}$ bases from $\mathcal{EL}^{\bot}_{\text{gfp}}$ bases for GCIs with high confidence, using the technique of unravelling $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions as discussed in Section 3.4.

The results thus obtained are again all effective, and we shall discuss some experiments in Section 5.3 that use the same data-set as the one used in Section 5.1. This allows us to directly compare the approaches of computing bases of valid GCIs on the one hand, and bases of GCIs with high confidence on the other. Moreover, we shall also discuss shortcomings of the approach of considering GCIs with high confidence, which will eventually lead us to considering extensions of the attribute exploration algorithm. These will be discussed in Chapter 6 and Chapter 7.

5.1. Computing Bases from DBpedia

We want to evaluate the practicability of the results of Baader and Distel by applying them to linked data extracted from the Linked Open Data Cloud. In other words, given some linked data, we want to extract a complete set of general concept inclusions that is valid within this data set. The goal of this experiment is to see in how far Baader and Distel's approach is practical in learning terminological knowledge about some domain, that is represented by linked data.

Of course, before we can do so we first have to discuss how we can obtain an interpretation from a given linked data set, and one which sufficiently reflects the logical structure of the initial data set.

Recall that a finite interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ over N_C and N_R consists of a set $\Delta^{\mathcal{I}}$ and a mapping $\cdot^{\mathcal{I}}$ that maps every $A \in N_C$ to a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and every $r \in N_R$ to a set of pairs

$r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. We have also already seen some examples of depicting interpretations as *graphs*, more precisely as *directed edge- and vertex-labeled graphs*. Indeed, interpretations are essentially nothing else than those graphs, where the set of vertex labels is N_C and the set of edge labels is N_R.

Linked data is quite similar to labeled graphs. More precisely, linked data is just an edge-labeled graph, represented by so-called *RDF-Triples* (where *RDF* stands for *Resource Description Framework*). Every triple consists of a *subject*, a *predicate*, and an *object* (in that order), each of them being an *uniform resource identifier* (URI). The idea is that RDF-Triples encode the information that the subject is connected to the object by means of the predicate. Two examples of RDF-Triples, taken from the DBpedia data set [25], are[1]

```
<http://dbpedia.org/resource/Aristotle>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://dbpedia.org/ontology/Philosopher> .

<http://dbpedia.org/resource/Aristotle>
<http://dbpedia.org/ontology/influenced>
<http://dbpedia.org/resource/Western_philosophy> .
```

Intuitively, these triples encode the facts that *Aristotle is (was) a philosopher* and that *Aristotle influenced Western Philosophy*.

Since RDF-Triples constitute edge-labeled graphs, we can take them as they are and regard them as interpretations over $N_C = \varnothing$ and N_R, where N_R is just the set of all predicates which appear in RDF-Triples in the data set. This approach would work, but it would only yield GCIs where no concept names are present. Such terminological knowledge may not be very interesting.

To alleviate this problem we make use of the special RDF predicate

```
http://www.w3.org/1999/02/22-rdf-syntax-ns#type        (5.1)
```

which expresses that a subject is an instance of a certain class.[2] If we consider triples with this predicates not as edges in the linked graph, but instead as the information that the subject is an instance of the object, then we indeed consider linked data as a vertex- and edge-labeled graph with vertex labels N_C and edge-labels N_R, where N_C is in general not empty.

This view on linked data now allows us to consider it as an interpretation over some sets N_C and N_R, and to apply Baader and Distel's results to such data sets. For our experiments, we have chosen a subset of the DBpedia data set as of March 2010 (Version 3.5)[3]. The linked data contained in the DBpedia data set has been extracted automatically from *Wikipedia*

[1] Indeed, these are *serializations* of RDF-Triples, in this case in the so-called *N-Triples* format

[2] `http://www.w3.org/1999/02/22-rdf-syntax-ns`

[3] `http://wiki.dbpedia.org/Downloads35?v=pb8`

Figure 5.1.: Wikipedia Article about Abraham Lincoln with Infobox on its Right

Infoboxes, which mean to represent facts about the topic of the current page in a compact way. An example of such an Infobox is shown in Figure 5.1.

The subset of the DBpedia data set we use for our experiments arises by restricting our attention to the relation

`http://dbpedia.org/ontology/child`.

From this subset, we constructed an interpretation $\mathcal{I}_{\text{DBpedia}} = (\Delta^{\mathcal{I}_{\text{DBpedia}}}, \cdot^{\mathcal{I}_{\text{DBpedia}}})$. To make the considerations easier to read, we shall drop the prefix `http://dbpedia.org/ontology` in the following, and just write child instead of `http://dbpedia.org/ontology/child`, for example.

To construct $\mathcal{I}_{\text{DBpedia}}$ we first compute all triples T_{child} from the DBpedia data set whose predicate is child. The subjects and objects of these triples are collected into $\Delta^{\mathcal{I}_{\text{DBpedia}}}$. We then define

$$\mathsf{child}^{\mathcal{I}_{\text{DBpedia}}} := \{\, (s, o) \in \Delta^{\mathcal{I}_{\text{DBpedia}}} \times \Delta^{\mathcal{I}_{\text{DBpedia}}} \mid (s, \mathsf{child}, o) \in T_{\mathsf{child}} \,\}.$$

Examples for triples contained in T_{child} are

```
<Abraham_Lincoln> <child> <Robert_Todd_Lincoln> .
<Abraham_Lincoln> <child> <Edward_Baker_Lincoln> .
```

Therefore,

$$(\mathsf{Abraham\_Lincoln}, \mathsf{Robert\_Todd\_Lincoln}) \in \mathsf{child}^{\mathcal{I}_{\text{DBpedia}}},$$
$$(\mathsf{Abraham\_Lincoln}, \mathsf{Edward\_Baker\_Lincoln}) \in \mathsf{child}^{\mathcal{I}_{\text{DBpedia}}}.$$

Then we consider all triples T_{type} whose predicate is the special type predicate of Equation (5.1) and whose subject is contained in $\Delta^{\mathcal{I}_{\text{DBpedia}}}$. The objects of those triples are collected

into a set N_C, i. e. they constitute the concept names of $\mathcal{I}_{\text{DBpedia}}$.[4] Then, for an $A \in N_C$, we define $A^{\mathcal{I}_{\text{DBpedia}}}$ to be the set of all subjects that appear in a triple in T_{type}, i. e.

$$A^{\mathcal{I}_{\text{DBpedia}}} := \{\, s \in \Delta^{\mathcal{I}_{\text{DBpedia}}} \mid (s, t, A) \in T_{\mathsf{type}} \,\}$$

where t stands for the special RDF type predicate of Equation (5.1). Example triples from T_{type} concerning `Abraham_Lincoln` are

```
<Abraham_Lincoln>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<Person> .

<Abraham_Lincoln>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<OfficeHolder> .
```

and therefore

$$\mathsf{Abraham\_Lincoln} \in \mathsf{Person}^{\mathcal{I}_{\text{DBpedia}}},$$
$$\mathsf{Abraham\_Lincoln} \in \mathsf{OfficeHolder}^{\mathcal{I}_{\text{DBpedia}}}.$$

The interpretation $\mathcal{I}_{\text{DBpedia}}$ then contains 5624 elements and 60 concept names, i. e. $|N_C| = 60$. By construction, there is only one role name in $\mathcal{I}_{\text{DBpedia}}$, namely child. To compute now bases of $\mathcal{I}_{\text{DBpedia}}$ means to extract all knowledge about the child-relation present in DBpedia and expressible in $\mathcal{EL}^\bot$. Note that elements from DBpedia are only present in $\mathcal{I}_{\text{DBpedia}}$ if they have children, or are children of someone else. Moreover, as DBpedia extracts its information from Wikipedia, all elements in $\mathcal{I}_{\text{DBpedia}}$ correspond to articles in Wikipedia; in particular, if such elements correspond to persons, then those persons have to be "sufficiently famous" in the sense that they deserve a Wikipedia article. Therefore, $\mathcal{I}_{\text{DBpedia}}$ represent DBpedia's knowledge about famous persons and their famous children.

We have to note, however, that the child-relation in DBpedia contains some false information, mostly due to the way this information is extracted from Wikipedia Infoboxes. More precisely, our interpretation $\mathcal{I}_{\text{DBpedia}}$ not only contains elements which correspond to humans, but also contains elements which are instances of Work, Organisation or PopulatedPlace, among others. However, those artifacts are comparably rare, and it is still reasonable to use $\mathcal{I}_{\text{DBpedia}}$ for our experiments.

We have implemented the algorithms devised by Baader and Distel to compute bases of finite interpretations[5] on top of `conexp-clj`[6], a general purpose tool for formal concept

[4] We omit `http://www.w3.org/2002/07/owl#Thing` in N_C, as it does not introduce any meaningful information.

[5] `http://github.com/exot/EL-exploration`

[6] `http://github.com/exot/conexp-clj`

analysis. When computing a minimal base of $\mathcal{I}_{\text{DBpedia}}$ as described in Theorem 4.3.8, we obtain a base $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ containing 1252 general concept inclusions. In the following we want to examine the GCIs contained in this base, describe some observations we made, and discuss the usefulness of these GCIs. Of course, we cannot do this formally, and shall therefore only argue intuitively.

Firstly, some of the GCIs contained in $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ constitute knowledge about the relationships among the concept names occurring in $\mathcal{I}_{\text{DBpedia}}$ only, for example

$$\begin{aligned} \mathsf{Politician} &\sqsubseteq \mathsf{Person} \\ \mathsf{MemberOfParliament} &\sqsubseteq \mathsf{Person} \sqcap \mathsf{Politician} \\ \mathsf{Criminal} \sqcap \mathsf{Politician} &\sqsubseteq \bot \end{aligned}$$

This knowledge can indeed be useful to learn the hierarchy of concept names (*taxonomy*) of $\mathcal{I}_{\text{DBpedia}}$. On the other hand, this does not yet show whether computing bases of $\mathcal{I}_{\text{DBpedia}}$ is useful, as the GCIs shown above could have easily been obtained by methods from formal concept analysis alone.

The last GCI states that there are no elements in $\mathcal{I}_{\text{DBpedia}}$ that are both criminal and politicians. GCIs of this form are called *disjointness constraints*, and they can be useful in applications. The disjointness constraints mentioned above contain only concept names. However, the approach by Baader and Distel enables us to find more disjointness constraints than just those between concept names. For example, $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ contains the GCI

$$\mathsf{Philosopher} \sqcap \exists \mathsf{child}.\top \sqsubseteq \bot$$

expressing that philosophers don't have children (or at least not children famous enough to occur in Wikipedia). Indeed, since $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ is complete for $\mathcal{I}_{\text{DBpedia}}$, all disjointness constraints valid in $\mathcal{I}_{\text{DBpedia}}$ can either be found in this base or are entailed by it. Such information can be of practical relevance.

There are other GCIs contained in $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ which are not disjointness constraints or only express knowledge about concept names. Two of them for which we could argue that they can be useful are

$$\begin{aligned} \exists \mathsf{child}.\mathsf{Person} &\sqsubseteq \mathsf{Person}, \\ \mathsf{FictionalCharacter} \sqcap \exists \mathsf{child}.\mathsf{Person} &\sqsubseteq \exists \mathsf{child}.\mathsf{FictionalCharacter} \end{aligned}$$

where the second GCI can be seen as a certain kind of $\mathcal{EL}^{\bot}$ approximation of the fact that fictional characters can only have fictional characters as their children. These GCIs can indeed be seen as useful terminological knowledge for the domain represented by $\mathcal{I}_{\text{DBpedia}}$.

Those GCIs, where one could say that they represent meaningful knowledge, are quite rare in $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$. On the other hand, $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ contains many GCIs whose usefulness is highly doubtful, either because they combine otherwise unrelated concepts, or they are too specific.

An example for the first case is

$$\mathsf{Person} \sqcap \exists \mathsf{child}.\mathsf{Book} \sqsubseteq \mathsf{FictionalCharacter}$$

which does not really represent any meaningful knowledge. On the other hand, this GCI indicates that DBpedia finds books as children only on Wikipedia pages on fictional characters. Thus, such GCIs may help to find errors in the way DBpedia extracts information, but are not helpful as knowledge themselves. Similar examples are

$$\begin{aligned} \exists \mathsf{child}.\mathsf{Newspaper} \sqcap \mathsf{Person} &\sqsubseteq \mathsf{Writer}, \\ \exists \mathsf{child}.\mathsf{MilitaryUnit} &\sqsubseteq \mathsf{Judge}, \\ \exists \mathsf{child}.\mathsf{Settlement} \sqcap \mathsf{Politician} &\sqsubseteq \mathsf{Congressman}. \end{aligned}$$

GCIs that can be considered as being too specific are the most common case among all GCIs in $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$, examples of them being

$$\begin{aligned} &\exists \mathsf{child}.\mathsf{Ambassador} \sqcap \mathsf{OfficeHolder} \\ &\sqsubseteq \exists \mathsf{child}.(\mathsf{Ambassador} \sqcap \exists \mathsf{child}.\mathsf{Person}) \sqcap \\ &\quad \exists \mathsf{child}.(\mathsf{OfficeHolder} \sqcap \exists \mathsf{child}.\mathsf{Congressman} \sqcap \exists \mathsf{child}.\mathsf{Actor} \sqcap \exists \mathsf{childOfficeHolder}). \end{aligned}$$

Indeed, this GCI only applies to the individual `Joseph_P._Kennedy%2C_Sr.`, and thus it states information only about this very individual and its children. We can handle such situations in principle by using *model exploration*, which allows for expert interaction in a similar way as attribute exploration does: if during model exploration the expert would encounter such a GCI as shown above, she would reject it as being too specific, and would provide counterexamples for it. We shall discuss this algorithm in more detail in Chapter 7.

Another approach to eliminating those over-specific GCIs could be to demand that the extracted GCIs apply to at least a certain number of different individuals. See Chapter 8 for more details on this.

Finally, among the GCIs contained in $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ there exist some which convey the impression of being redundant, like

$$\exists \mathsf{child}.\exists \mathsf{child}.\top \sqsubseteq \exists \mathsf{child}.(\mathsf{Person} \sqcap \exists \mathsf{child}.\top),$$

because it should be quite clear from the construction of $\mathcal{I}_{\text{DBpedia}}$ that only persons can have children, i. e. the following GCI

$$\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person} \tag{5.2}$$

should hold in $\mathcal{I}_{\text{DBpedia}}$. The reason for that is that only Wikipedia articles of human beings should contain a child-entry within their Infobox. Thus, even with all the errors in $\mathcal{I}_{\text{DBpedia}}$ concerning non-persons that we have discussed before, the GCI in Equation (5.2) should hold in $\mathcal{I}_{\text{DBpedia}}$.

However, this is not the case, since there are four counterexamples to this GCI in $\mathcal{I}_{\text{DBpedia}}$, i. e. elements $x \in \Delta^{\mathcal{I}_{\text{DBpedia}}}$ satisfying $x \in (\exists \mathsf{child}.\top)^{\mathcal{I}_{\text{DBpedia}}} \backslash \mathsf{Person}^{\mathcal{I}_{\text{DBpedia}}}$. These are[7]

`Teresa_Carpio, Charles_Heung, Adam_Cheng, Lydia_Shum.`

However, these elements represent human beings, so they *should* actually be instances of Person. In other words, these counterexamples are all *erroneous* counterexamples, but since they are present in $\mathcal{I}_{\text{DBpedia}}$, the approach developed by Baader and Distel will not ignore them. This not only inhibits finding the GCI $\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}$, but also causes incomprehensible GCIs to be found, which are special cases of this GCI but somehow try to "circumvent" the erroneous counterexamples. An example for this is

$$\begin{aligned}&\mathsf{Person} \sqcap \exists \mathsf{child}.(\mathsf{Person} \sqcap \exists \mathsf{child}.(\mathsf{Person} \sqcap \exists \mathsf{child}.(\mathsf{Person} \sqcap \\ &\quad \exists \mathsf{child}.\exists \mathsf{child}.(\mathsf{Person} \sqcap \exists \mathsf{child}.\top)))) \\ &\sqsubseteq \exists \mathsf{child}.(\mathsf{Person} \sqcap \exists \mathsf{child}.(\mathsf{Person} \sqcap \exists \mathsf{child}.(\mathsf{Person} \sqcap \exists \mathsf{child}.(\mathsf{Person} \sqcap \\ &\quad \exists \mathsf{child}.\exists \mathsf{child}.\mathsf{Person})))).\end{aligned}$$

5.2. GCIs with High Confidence in Finite Interpretations

It would certainly increase the applicability of Baader and Distel's approach if we could ignore erroneous counterexamples in our data, as then we could extract simpler and more general GCIs from this data. However, we cannot expect to have an automatic procedure which achieves this goal, i. e. we cannot expect an algorithm that automatically ignores erroneous counterexamples. The reason for that is that the algorithm would need to know how to distinguish errors in the current domain of interest, and for this the algorithm would need to possess knowledge about this domain, which however we are just about to learn.

On the other hand, we can assume is our initially given interpretation $\mathcal{I}$ contains only *few* errors, as otherwise learning GCIs from it would be futile. Based on this assumption we can approach the problem of erroneous counterexamples as follows: in the case that $\mathcal{I}$ contains much more *positive examples* for a GCI $C \sqsubseteq D$ than *negative* ones, we assume that the negative counterexamples are "probably erroneous." Here, a *positive example* for $C \sqsubseteq D$ would be an element $x \in \Delta^{\mathcal{I}}$ satisfying $x \in C^{\mathcal{I}} \cap D^{\mathcal{I}}$, and a *negative example* for $C \sqsubseteq D$ would be an element $y \in \Delta^{\mathcal{I}}$ such that $y \in C^{\mathcal{I}} \backslash D^{\mathcal{I}}$. We can consider a GCI to be "almost valid" in $\mathcal{I}$ if the number of positive examples is much higher than the number of negative ones. The approach to ignore erroneous counterexamples would then consider these "almost valid" GCIs in addition to the valid ones, and try to find finite bases for both of them.

[7] Coincidentally, all these are artists from Hong Kong.

This actually works quite well for our example interpretation $\mathcal{I}_{\text{DBpedia}}$, as $\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}$ has much more positive than negative examples: there are 2551 elements $x \in \Delta^{\mathcal{I}_{\text{DBpedia}}}$ to which $\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}$ applies, i. e. they satisfy $x \in (\exists \mathsf{child}.\top)^{\mathcal{I}_{\text{DBpedia}}}$, but only 4 of those (the ones mentioned above) fail to also satisfy $x \in \mathsf{Person}^{\mathcal{I}_{\text{DBpedia}}}$. Therefore, $\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}$ has 2547 positive examples, but only 4 negative ones. By our approach, we can consider these negative examples as errors and ignore them. Thus, $\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}$ would be extracted from $\mathcal{I}_{\text{DBpedia}}$.

Of course, this approach is highly heuristic: if valid counterexamples for a GCI $C \sqsubseteq D$ are just *rare* in $\mathcal{I}$, then the above sketched approach would treat them as errors, which is incorrect. On the other hand, it may be much more desirable to extract GCIs which are *wrong* in some application domain, than to miss GCIs which are *correct*, as long as not too many wrong GCIs are being extracted. This is because identifying wrong GCIs can be considered much easier than finding correct GCIs that are just invalidated by errors in the data.

It is the purpose of this section to give a formalization of the notion of a GCI to be "almost valid" in some finite interpretation. We shall base this formalization on the notion of *confidence* as it is used in data-mining [1]. Our goal is then to find *finite bases* of all GCIs which enjoy a *high confidence* in the initially given interpretation.

5.2.1. Confidence of GCIs and Confident Bases

We argued intuitively that the number of positive examples should be "much higher" than the number of negative examples. To formalize this notion, we define the *confidence* of $C \sqsubseteq D$ in $\mathcal{I}$ as follows.

5.2.1 Definition (Confidence of GCIs) Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation over N_C and N_R, and let $C, D \in \mathcal{EL}^{\bot}_{\text{gfp}}(N_C, N_R)$. Then the *confidence* of $C \sqsubseteq D$ in $\mathcal{I}$, written as $\text{conf}_{\mathcal{I}}(C \sqsubseteq D)$, is defined as

$$\text{conf}_{\mathcal{I}}(C \sqsubseteq D) = \begin{cases} 1 & C^{\mathcal{I}} = \varnothing \\ \frac{|(C \sqcap D)^{\mathcal{I}}|}{|C^{\mathcal{I}}|} & \text{otherwise.} \end{cases} \qquad \diamondsuit$$

Note that $(C \sqcap D)^{\mathcal{I}}$ is just the number of positive examples for $C \sqsubseteq D$ in $\mathcal{I}$, and that $|C^{\mathcal{I}}|$ is just the number of all elements to which $C \sqsubseteq D$ applies, i. e. the number of all positive and negative examples. Therefore, $\text{conf}_{\mathcal{I}}(C \sqsubseteq D)$ measures the amount of positive examples against the number of all elements to which $C \sqsubseteq D$ applies: the higher the confidence of $C \sqsubseteq D$ in $\mathcal{I}$, the more the number of positive examples is higher than the number of negative ones.

5.2.2 Example Recall that $\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}$ has 2547 positive and 4 negative examples in $\mathcal{I}_{\text{DBpedia}}$, thus

$$\text{conf}_{\mathcal{I}_{\text{DBpedia}}}(\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}) = \frac{2547}{2551} \approx 0.998. \qquad \diamondsuit$$

To use the confidence of GCIs in finite interpretations to formalize the notion of being "almost true" we need to choose a *threshold* above which we can say that the number of positive examples is "much higher" than the number of negative ones.

5.2.3 Definition (Confidence-Based Theory of Finite Interpretations) Let $\mathcal{I}$ be a finite interpretation over N_C and N_R, and let $c \in [0,1]$. Then the *confidence-based theory* $\mathrm{Th}_c(\mathcal{I})$ of $\mathcal{I}$ *with threshold* c is defined as

$$\mathrm{Th}_c(\mathcal{I}) := \{\, C \sqsubseteq D \mid C, D \in \mathcal{EL}^{\bot}_{\mathrm{gfp}}(N_C, N_R), \mathrm{conf}_{\mathcal{I}}(C \sqsubseteq D) \geqslant c \,\}.$$

We say that GCIs in $\mathrm{Th}_c(\mathcal{I})$ have *high confidence* (with respect to the threshold c). Sometimes, we may call the elements of $\mathrm{Th}_c(\mathcal{I})$ just *confident GCIs*. ♢

Note that in general the set $\mathrm{Th}_c(\mathcal{I})$ is not closed under entailment.

Note that $C \sqsubseteq D$ has confidence 1 in $\mathcal{I}$ if and only if $C \sqsubseteq D$ holds in $\mathcal{I}$. Therefore, $\mathrm{Th}(\mathcal{I}) \subseteq \mathrm{Th}_c(\mathcal{I})$, and thus the set $\mathrm{Th}_c(\mathcal{I})$ is also infinite in general (i. e. when $N_R \neq \varnothing$). Thus, again we want to find *finite bases*, this time of the set $\mathrm{Th}_c(\mathcal{I})$. Recall that a set $\mathcal{B}$ of general concept inclusions is a base of $\mathrm{Th}_c(\mathcal{I})$ if it is sound and complete for $\mathrm{Th}_c(\mathcal{I})$, i. e. if all GCIs in $\mathcal{B}$ are entailed by $\mathrm{Th}_c(\mathcal{I})$ and vice versa.

Notice, however, that since $\mathrm{Th}_c(\mathcal{I})$ is not closed under entailment it may happen that $\mathcal{B} \nsubseteq \mathrm{Th}_c(\mathcal{I})$. On the other hand, it may be desirable to have bases $\mathcal{B}$ of $\mathrm{Th}_c(\mathcal{I})$ where the GCIs contained in this base have high confidence as well, i. e. they satisfy $\mathcal{B} \subseteq \mathrm{Th}_c(\mathcal{I})$. We shall call such bases *confident bases* of $\mathrm{Th}_c(\mathcal{I})$.

5.2.2. Luxenburger's Base

Before we consider finite bases and finite confident bases of $\mathrm{Th}_c(\mathcal{I})$, let us first consider the analogous problem in formal concept analysis, which is to find small bases of *implications with high confidence* in a given formal context. The relevant results of this discussion mainly go back to results by Luxenburger on *partial implications* [68, 69], and extensions thereof [97]. We shall first discuss these results, however not in their original form, but instead in a way suitable for our considerations.

We have already defined the notion of confidence of a GCI in some given finite interpretation. The analogous notion in formal concept analysis is the confidence of an implication in a formal context.

5.2.4 Definition (Confidence of Implications) Let $\mathbb{K} = (G, M, I)$ be a finite formal context, and let $(A \to B) \in \mathrm{Imp}(M)$. Then the *confidence* of $A \to B$ in $\mathbb{K}$ is defined as

$$\mathrm{conf}_{\mathbb{K}}(A \to B) := \begin{cases} 1 & A' = \varnothing \\ \frac{|(A \cup B)'|}{|A'|} & \text{otherwise.} \end{cases}$$ ♢

We can now consider an implication to have *high confidence* in a formal context if and only if it is above a certain threshold $c \in [0,1]$.

5.2.5 Definition (Confidence-Based Theory of Finite Formal Contexts) Let $\mathbb{K} = (G, M, I)$ be a finite formal context, and let $c \in [0, 1]$. The *confidence-based theory* of $\mathbb{K}$ *with threshold* c is defined as

$$\mathrm{Th}_c(\mathbb{K}) := \{ (A \to B) \in \mathrm{Imp}(M) \mid \mathrm{conf}_{\mathbb{K}}(A \to B) \geqslant c \}.$$

An implication $(A \to B) \in \mathrm{Imp}(M)$ is said to have *high confidence* in $\mathbb{K}$ if and only if $(A \to B) \in \mathrm{Th}_c(\mathbb{K})$. Sometimes, those implications are also called *confident implications*.$\diamondsuit$

We now want to find small bases of $\mathrm{Th}_c(\mathbb{K})$ for finite formal contexts $\mathbb{K} = (G, M, I)$ and $c \in [0, 1]$. Of course, the term "small" is rather subjective here, but recall that the main purpose of this discussion is to obtain a finite base of $\mathrm{Th}_c(\mathbb{K})$ which, when transferred to the side of description logics, stays finite. The main idea in that direction is that, if an interpretation $\mathcal{I}$ is finite, then it only has finitely many model-based most-specific concept descriptions (up to equivalence). Since model-based most-specific concept descriptions are in a one-to-one correspondence to the intents of the induced context $\mathbb{K}_{\mathcal{I}}$ of $\mathcal{I}$, finding a base of $\mathrm{Th}_c(\mathbb{K})$ which only contains intents of $\mathbb{K}$ is likely suitable for our purpose.

To compute such a base of $\mathrm{Th}_c(\mathbb{K})$ we first observe that, as in the case of GCIs, an implication $(A \to B)$ holds in $\mathbb{K}$ if and only if its confidence in $\mathbb{K}$ is 1. Therefore, $\mathrm{Th}(\mathbb{K}) \subseteq \mathrm{Th}_c(\mathbb{K})$ is true for all $c \in [0, 1]$. Furthermore, we already know how to compute bases of $\mathrm{Th}(\mathbb{K})$. Thus, if $\mathcal{L}$ is a base of $\mathrm{Th}(\mathbb{K})$, to find a base of $\mathrm{Th}_c(\mathbb{K})$ it is enough to compute bases of $\mathrm{Th}_c(\mathbb{K})$ with background knowledge $\mathcal{L}$.

The first crucial observation now is the well-known fact that

$$\mathrm{conf}_{\mathbb{K}}(A \to B) = \mathrm{conf}_{\mathbb{K}}(A'' \to B'')$$

is always true for $(A \to B) \in \mathrm{Imp}(M)$, because $A''' = A'$ and

$$\begin{aligned}(A \cup B)' &= A' \cap B' \\ &= A''' \cap B''' \\ &= (A'' \cup B'')'.\end{aligned}$$

But then $\{ A'' \to B'' \} \models (A \to B)$, so it is enough to consider only implications where both premise and conclusion are intents of $\mathbb{K}$. The following lemma makes use of this observation.

5.2.6 Lemma *Let $\mathbb{K} = (G, M, I)$ be a formal context, let $c \in [0, 1]$ and let $\mathcal{L}$ be a base of $\mathbb{K}$. Then the set*

$$\mathrm{Conf}(\mathbb{K}, c) := \{ A'' \to B'' \mid A \subseteq B \subseteq M, 1 > \mathrm{conf}_{\mathbb{K}}(A'' \to B'') \geqslant c \}$$

is a confident base of $\mathrm{Th}_c(\mathbb{K})$ *with background knowledge* $\mathcal{L}$.

Proof We first observe that for $A, B \subseteq M$ it is true that

$$\{ A'' \to (A \cup B)'' \} \models (A'' \to B''),$$

because $B'' \subseteq (A \cup B)''$. Then if $(X \to Y) \in \mathrm{Th}_c(\mathbb{K})$, then

$$\mathcal{L} \models (X \to X''),$$

and thus

$$\mathcal{L} \cup \{ X'' \to (X \cup Y)'' \} \models (X \to Y).$$

Furthermore,

$$\mathrm{conf}_{\mathbb{K}}(X'' \to (X \cup Y)'') = \mathrm{conf}_{\mathbb{K}}(X \to Y) \geqslant c,$$

therefore $(X'' \to (X \cup Y)'') \in \mathrm{Conf}(\mathbb{K}, c)$. Thus

$$\mathcal{L} \cup \mathrm{Conf}(\mathbb{K}, c) \models (X \to Y)$$

and therefore $\mathrm{Conf}(\mathbb{K}, c)$ is a base of $\mathrm{Th}_c(\mathbb{K})$ with background knowledge $\mathcal{L}$. □

The set $\mathrm{Conf}(\mathbb{K}, c)$ can be further reduced by the following, well-known observation.

5.2.7 Lemma *Let $\mathbb{K} = (G, M, I)$ be a finite formal context, and let $A \subseteq B \subseteq C \subseteq M$. Then*

$$\mathrm{conf}_{\mathbb{K}}(A \to C) = \mathrm{conf}_{\mathbb{K}}(A \to B) \cdot \mathrm{conf}_{\mathbb{K}}(B \to C). \tag{5.3}$$

Proof If $A' = \emptyset$, then $B' = C' = \emptyset$, and both sides of the equation are 1. If $A' \neq \emptyset$ but $B' = \emptyset$, then $C' = \emptyset$ and both sides of the equation are 0. In both cases, equality holds.

Now let $A' \neq \emptyset \neq B'$. Then we can easily compute

$$\begin{aligned} \mathrm{conf}_{\mathbb{K}}(A \to C) &= \frac{|(A \cup C)'|}{|A'|} \\ &= \frac{|(A \cup B)'|}{|A'|} \cdot \frac{|(A \cup C)'|}{|(A \cup B)'|} \end{aligned}$$

and since $A \subseteq B \subseteq C$ it is true that $A \cup B = B$ and $A \cup C = C = B \cup C$, thus we can continue

$$\begin{aligned} \mathrm{conf}_{\mathbb{K}}(A \to C) &= \frac{|(A \cup B)'|}{|A'|} \cdot \frac{|(B \cup C)'|}{|B'|} \\ &= \mathrm{conf}_{\mathbb{K}}(A \to B) \cdot \mathrm{conf}_{\mathbb{K}}(B \to C) \end{aligned}$$

and the claim is proven. □

A simple consequence of this lemma is that if $A'' \subsetneq B'' \subsetneq C''$ and $(A'' \to C'') \in \mathrm{Conf}(\mathbb{K}, c)$, then $(A'' \to B''), (B'' \to C'') \in \mathrm{Conf}(\mathbb{K}, c)$ and since

$$\{ A'' \to B'', B'' \to C'' \} \models (A'' \to C'')$$

the implication $A'' \to C''$ is dispensable in $\mathrm{Conf}(\mathbb{K}, c)$, and can thus be removed without any harm.

5.2.8 Theorem *Let $\mathbb{K} = (G, M, I)$ be a finite formal context, and let $c \in [0, 1]$. Let $\mathcal{L}$ be a base of $\mathbb{K}$. Then*

$$\mathrm{Lux}(\mathbb{K}, c) := \{ (A'' \to C'') \mid A \subseteq C \subseteq M, 1 > \mathrm{conf}_{\mathbb{K}}(A'' \to C'') \geqslant c, \nexists B \subseteq M\colon A'' \subsetneq B'' \subsetneq C'' \}$$

is a confident base of $\mathrm{Th}_c(\mathbb{K})$ with background knowledge $\mathcal{L}$.

The proof of this theorem is inspired by a similar proof from [97].

Proof Let $(A'' \to C'') \in \mathrm{Conf}(\mathbb{K}, c)$. Then it is enough to show that

$$\mathrm{Lux}(\mathbb{K}, c) \models (A'' \to C'').$$

Because $(A'' \to C'') \in \mathrm{Conf}(\mathbb{K}, c)$ it is true that $A'' \subsetneq C''$. Since $\mathbb{K}$ is finite, the lattice of all intents of $\mathbb{K}$ is finite, as well. Therefore, there exists a chain of intents

$$B_0 = A'' \subsetneq B_1 \subsetneq B_2 \subsetneq \cdots \subsetneq B_{n-1} \subsetneq B_n = C''$$

such that for all $i \in \{ 0, \ldots, n-1 \}$ there is no intent D satisfying $B_i \subsetneq D \subsetneq B_{i+1}$. Then, by induction, Lemma 5.2.7 yields

$$\mathrm{conf}_{\mathbb{K}}(A'' \to C'') = \prod_{i=0}^{n-1} \mathrm{conf}_{\mathbb{K}}(B_i \to B_{i+1}).$$

Since $\mathrm{conf}_{\mathbb{K}}(B_i \to B_{i+1}) \in [0, 1]$, this yields

$$\mathrm{conf}_{\mathbb{K}}(B_i \to B_{i+1}) \geqslant \mathrm{conf}_{\mathbb{K}}(A'' \to C'') \geqslant c$$

and therefore $(B_i \to B_{i+1}) \in \mathrm{Lux}(\mathbb{K}, c)$ for $i \in \{ 0, \ldots, n-1 \}$. Thus,

$$\mathrm{Lux}(\mathbb{K}, c) \models (A'' \to C'')$$

as required. □

Clearly we can weaken the prerequisites of this theorem as follows: instead of considering the whole set $\mathrm{Lux}(\mathbb{K}, c)$, a subset $\mathcal{B} \subseteq \mathrm{Lux}(\mathbb{K}, c)$ that is complete for $\mathrm{Lux}(\mathbb{K}, c)$ is just sufficient. Moreover, it is not necessary to consider only bases $\mathcal{L}$ of $\mathbb{K}$. Instead, it is sufficient to take a set $\mathcal{L} \subseteq \mathrm{Th}(\mathbb{K})$ such that $\mathcal{L} \cup \mathcal{B}$ is complete for $\mathbb{K}$.

5.2.9 Corollary *Let $\mathbb{K} = (G, M, I)$ be a finite formal context, and let $c \in [0, 1]$. If $\mathcal{B} \subseteq \mathrm{Th}_c(\mathbb{K})$ is complete for $\mathrm{Lux}(\mathbb{K}, c)$, and if $\mathcal{L} \subseteq \mathrm{Th}(\mathbb{K})$ satisfies that $\mathcal{L} \cup \mathcal{B}$ is complete for $\mathrm{Th}(\mathbb{K})$, then $\mathcal{B}$ is a confident base of $\mathrm{Th}_c(\mathbb{K})$ with background knowledge $\mathcal{L}$.*

5.2.3. A Luxenburger-Style Base of all GCIs with High Confidence

We can now use the results about confident bases of implications with high confidence to obtain confident bases of GCIs with high confidence. For this, we simply adapt the results of the previous section and prove them again in the setting of description logics. Notice that because of this, the following section is very similar to the previous one.

We start with the observation that the confidence of GCIs does not change if we switch to model-based most-specific concept descriptions.

5.2.10 Lemma *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation over N_C and N_R, and let C and D be $\mathcal{EL}^{\bot}_{gfp}$ concept descriptions over N_C and N_R. Then*

$$\mathrm{conf}_{\mathcal{I}}(C \sqsubseteq D) = \mathrm{conf}_{\mathcal{I}}(C^{\mathcal{I}\mathcal{I}} \sqsubseteq D^{\mathcal{I}\mathcal{I}}).$$

Proof The idea is the same as in the case of confidence of implications, just with a different notation.

Since $C^{\mathcal{I}} = C^{\mathcal{I}\mathcal{I}\mathcal{I}}$, we have that $C^{\mathcal{I}} = \varnothing$ if and only if $C^{\mathcal{I}\mathcal{I}\mathcal{I}} = \varnothing$. In this case, both sides of the equation are 1 and equality holds.

Let $C^{\mathcal{I}} \neq \varnothing$. Then $C^{\mathcal{I}\mathcal{I}\mathcal{I}} \neq \varnothing$, and we can compute

$$\begin{aligned}
\mathrm{conf}_{\mathcal{I}}(C \sqsubseteq D) &= \frac{|(C \sqcap D)^{\mathcal{I}}|}{|C^{\mathcal{I}}|} \\
&= \frac{|C^{\mathcal{I}} \cap D^{\mathcal{I}}|}{|C^{\mathcal{I}}|} \\
&= \frac{|C^{\mathcal{I}\mathcal{I}\mathcal{I}} \cap D^{\mathcal{I}\mathcal{I}\mathcal{I}}|}{|C^{\mathcal{I}\mathcal{I}\mathcal{I}}|} \\
&= \mathrm{conf}_{\mathcal{I}}(C^{\mathcal{I}\mathcal{I}} \sqsubseteq D^{\mathcal{I}\mathcal{I}})
\end{aligned}$$

as required. □

From this fact we can now derive the analog of Lemma 5.2.7.

5.2.11 Theorem *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation over N_C and N_R, and let $\mathcal{B}$ be a finite base of $\mathcal{I}$. Let $c \in [0,1]$ and define*

$$\mathrm{Conf}(\mathcal{I}, c) := \{\, X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}} \mid Y \subseteq X \subseteq \Delta^{\mathcal{I}}, 1 > \mathrm{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \geqslant c \,\}.$$

Then the set $\mathrm{Conf}(\mathcal{I}, c) \cup \mathcal{B}$ is a finite confident base of $\mathrm{Th}_c(\mathcal{I})$.

In the definition of $\mathrm{Conf}(\mathcal{I}, c)$ we consider of course all GCIs only up to equivalence: if $(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}), (\overline{X}^{\mathcal{I}} \sqsubseteq \overline{Y}^{\mathcal{I}}) \in \mathrm{Conf}(\mathcal{I}, c)$ are such that $X^{\mathcal{I}} \equiv \overline{X}^{\mathcal{I}}, Y^{\mathcal{I}} \equiv \overline{Y}^{\mathcal{I}}$, we only keep one of these GCIs in $\mathrm{Conf}(\mathcal{I}, c)$, and discard the other one.

Proof Clearly, $\mathrm{Conf}(\mathcal{I}, c) \cup \mathcal{B} \subseteq \mathrm{Th}_c(\mathcal{I})$. Furthermore, since $\Delta^{\mathcal{I}}$ is finite, $\mathrm{Conf}(\mathcal{I}, c)$ is finite as well. Thus, it remains to show that $\mathrm{Conf}(\mathcal{I}, c) \cup \mathcal{B}$ is complete for $\mathrm{Th}_c(\mathcal{I})$.

Let $(C \sqsubseteq D) \in \mathrm{Th}_c(\mathcal{I})$. If $C \sqsubseteq D$ is valid in $\mathcal{I}$, then it is entailed by $\mathcal{B}$, and nothing remains to be shown.

Therefore, let $C \sqsubseteq D$ be not valid in $\mathcal{I}$. Then observe that $C \sqsubseteq D$ is entailed by $C \sqsubseteq C \sqcap D$. Furthermore, $\mathcal{B}$ entails $C \sqsubseteq C^{\mathcal{II}}$, and thus

$$\mathrm{conf}_{\mathcal{I}}(C \sqsubseteq D) = \mathrm{conf}_{\mathcal{I}}(C \sqsubseteq C \sqcap D) = \mathrm{conf}_{\mathcal{I}}(C^{\mathcal{II}} \sqsubseteq (C \sqcap D)^{\mathcal{II}})$$

by Lemma 5.2.10, and since $(C \sqsubseteq D) \in \mathrm{Th}_c(\mathcal{I})$ we obtain

$$(C^{\mathcal{II}} \sqsubseteq (C \sqcap D)^{\mathcal{II}}) \in \mathrm{Conf}(\mathcal{I}, c)$$

by choosing $X = C^{\mathcal{I}}$ and $Y = (C \sqcap D)^{\mathcal{I}}$. But then

$$\mathrm{Conf}(\mathcal{I}, c) \cup \mathcal{B} \models (C \sqsubseteq C^{\mathcal{II}}), (C^{\mathcal{II}} \sqsubseteq (C \sqcap D)^{\mathcal{II}})$$

and since $(C \sqcap D)^{\mathcal{II}} \sqsubseteq D^{\mathcal{II}} \sqsubseteq D$, it follows that

$$\mathrm{Conf}(\mathcal{I}, c) \cup \mathcal{B} \models (C \sqsubseteq D)$$

as required. □

It is also possible to establish the analog to Lemma 5.2.7.

5.2.12 Lemma *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation, and let $Z \subseteq Y \subseteq X$. Then*

$$\mathrm{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Z^{\mathcal{I}}) = \mathrm{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \cdot \mathrm{conf}_{\mathcal{I}}(Y^{\mathcal{I}} \sqsubseteq Z^{\mathcal{I}}).$$

Proof If $X^{\mathcal{II}} = \emptyset$, then because of $Z \subseteq Y \subseteq X$ we obtain $Z^{\mathcal{II}} \subseteq Y^{\mathcal{II}} \subseteq X^{\mathcal{II}}$ and thus $Z^{\mathcal{II}} = Y^{\mathcal{II}} = \emptyset$. In this case, both sides of the equation are 1.

If $X^{\mathcal{II}} \neq \emptyset$ but $Y^{\mathcal{II}} = \emptyset$, then $Z^{\mathcal{II}} = \emptyset$ and both sides of the equation are 0. In both cases, equality holds.

Let $X^{\mathcal{I}\mathcal{I}} \neq \varnothing \neq Y^{\mathcal{I}\mathcal{I}}$. As in the case of Lemma 5.2.7 we can compute

$$\begin{aligned}\operatorname{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Z^{\mathcal{I}}) &= \frac{|(X^{\mathcal{I}} \sqcap Z^{\mathcal{I}})^{\mathcal{I}}|}{|X^{\mathcal{I}\mathcal{I}}|} \\ &= \frac{|(X^{\mathcal{I}} \sqcap Y^{\mathcal{I}})^{\mathcal{I}}|}{|X^{\mathcal{I}\mathcal{I}}|} \cdot \frac{|(X^{\mathcal{I}} \sqcap Z^{\mathcal{I}})^{\mathcal{I}}|}{|(X^{\mathcal{I}} \sqcap Y^{\mathcal{I}})^{\mathcal{I}}|} \\ &= \frac{|(X^{\mathcal{I}} \sqcap Y^{\mathcal{I}})^{\mathcal{I}}|}{|X^{\mathcal{I}\mathcal{I}}|} \cdot \frac{|(Y^{\mathcal{I}} \sqcap Z^{\mathcal{I}})^{\mathcal{I}}|}{|Y^{\mathcal{I}\mathcal{I}}|} \\ &= \operatorname{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \cdot \operatorname{conf}_{\mathcal{I}}(Y^{\mathcal{I}} \sqsubseteq Z^{\mathcal{I}})\end{aligned}$$

because $Z^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}} \sqsubseteq X^{\mathcal{I}}$. □

5.2.13 Theorem *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation, and let $c \in [0,1]$. Let $\mathcal{B}$ be a base of $\mathcal{I}$. Define*

$$\begin{aligned}\operatorname{Lux}(\mathcal{I}, c) := \{\, X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}} \mid Y \subseteq X \subseteq \Delta^{\mathcal{I}}, \\ 1 > \operatorname{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \geqslant c, \nexists Z \subseteq \Delta^{\mathcal{I}} \colon Y^{\mathcal{I}} \subsetneq Z^{\mathcal{I}} \subsetneq X^{\mathcal{I}} \,\}.\end{aligned}$$

Then $\operatorname{Lux}(\mathcal{I}, c) \cup \mathcal{B}$ is a finite confident base of $\operatorname{Th}_c(\mathcal{I})$.

Proof The proof is again analogous to the one of the corresponding Theorem 5.2.8. To show the claim it is sufficient to just show that all GCIs in $\operatorname{Conf}(\mathcal{I}, c)$ are entailed by $\operatorname{Lux}(\mathcal{I}, c)$. To this end, let $(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \in \operatorname{Conf}(\mathcal{I}, c)$. Then $Y \subseteq X \subseteq \Delta^{\mathcal{I}}$, i. e. $Y^{\mathcal{I}} \sqsubseteq X^{\mathcal{I}}$. Since $\Delta^{\mathcal{I}}$ is finite, there exist sets $\Delta^{\mathcal{I}} \supseteq Z_0 \supseteq Z_1 \supseteq \cdots \supseteq Z_n$ satisfying

$$Y^{\mathcal{I}} = Z_n^{\mathcal{I}} \subsetneq Z_{n-1}^{\mathcal{I}} \subsetneq \cdots \subsetneq Z_1^{\mathcal{I}} \subsetneq Z_0^{\mathcal{I}} = X^{\mathcal{I}}$$

such that there do not exist sets $W \subseteq \Delta^{\mathcal{I}}$ with

$$Z_i^{\mathcal{I}} \subsetneq W^{\mathcal{I}} \subsetneq Z_{i-1}^{\mathcal{I}} \tag{5.4}$$

for any $i \in \{1, \ldots, n\}$. Then by Lemma 5.2.12 it is true that

$$\operatorname{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) = \prod_{i=0}^{n-1} \operatorname{conf}_{\mathcal{I}}(Z_i^{\mathcal{I}} \sqsubseteq Z_{i+1}^{\mathcal{I}}).$$

Since the confidence is always an element of $[0,1]$, we obtain from this equality that

$$\operatorname{conf}_{\mathcal{I}}(Z_i^{\mathcal{I}} \sqsubseteq Z_{i+1}^{\mathcal{I}}) \geqslant \operatorname{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \geqslant c$$

and because of Equation (5.4) we obtain $(Z_i^{\mathcal{I}} \sqsubseteq Z_{i+1}^{\mathcal{I}}) \in \operatorname{Lux}(\mathcal{I}, c)$ for all $i \in \{0, \ldots, n-1\}$. Since $\{\, Z_i^{\mathcal{I}} \sqsubseteq Z_{i+1}^{\mathcal{I}} \mid i \in \{0, \ldots, n-1\} \,\}$ entails $X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}$ we obtain

$$\operatorname{Lux}(\mathcal{I}, c) \models (X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}})$$

as required. □

And finally, the analog of Corollary 5.2.9 of course holds as well.

5.2.14 Corollary *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation, and let $c \in [0,1]$. If $\mathcal{B} \subseteq \mathrm{Th}_c(\mathcal{I})$ is complete for $\mathrm{Lux}(\mathbb{K}, c)$, and if $\mathcal{L} \subseteq \mathrm{Th}(\mathcal{I})$ is such that $\mathcal{B} \cup \mathcal{L}$ is complete for $\mathrm{Th}(\mathcal{I})$, then $\mathcal{B} \cup \mathcal{L}$ is a confident base of $\mathrm{Th}_c(\mathcal{I})$.*

To compute the sets $\mathrm{Conf}(\mathcal{I}, c)$ and $\mathrm{Lux}(\mathcal{I}, c)$ we can just compute all model-based most-specific concept descriptions, and compute for each two $X^{\mathcal{I}}, Y^{\mathcal{I}}$, $Y \subseteq X \subseteq \Delta^{\mathcal{I}}$ whether

$$\mathrm{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \geqslant c.$$

However, we can transfer the computation of these sets into a computation which can solely be done in the induced formal context $\mathbb{K}_{\mathcal{I}}$. This may be desirable because the computations in $\mathbb{K}_{\mathcal{I}}$ may be easier to conduct, since we only have to work with subsets of $M_{\mathcal{I}}$, and not with complex concept descriptions.

The actual transformation is quite simple: it is true that

$$\begin{aligned} \mathrm{Conf}(\mathcal{I}, c) &= \bigsqcap \mathrm{Conf}(\mathbb{K}_{\mathcal{I}}, c) \\ \mathrm{Lux}(\mathcal{I}, c) &= \bigsqcap \mathrm{Lux}(\mathbb{K}_{\mathcal{I}}, c) \end{aligned} \tag{5.5}$$

where the equality is meant up to equivalence, i. e. every GCI in the set on the left-hand side is equivalent to one in the right-hand side, and vice versa.

Establishing these equations is also not difficult. We start with a simple connection between the confidence of GCIs in $\mathcal{I}$ and the confidence of implications in $\mathbb{K}_{\mathcal{I}}$.

5.2.15 Proposition *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation, and let $X, Y \subseteq \Delta^{\mathcal{I}}$. Then*

$$\mathrm{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) = \mathrm{conf}_{\mathbb{K}_{\mathcal{I}}}(X' \to Y').$$

Proof By Proposition 4.2.10 it is true that $X^{\mathcal{I}\mathcal{I}} = X''$. Thus, if $X^{\mathcal{I}\mathcal{I}} = \varnothing$, then $X'' = \varnothing$ and

$$\mathrm{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) = 1 = \mathrm{conf}_{\mathbb{K}_{\mathcal{I}}}(X' \to Y').$$

If $X^{\mathcal{I}\mathcal{I}} \neq \varnothing$, then $X'' \neq \varnothing$ and we can compute

$$\begin{aligned} \mathrm{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) &= \frac{|(X^{\mathcal{I}} \sqcap Y^{\mathcal{I}})^{\mathcal{I}}|}{|X^{\mathcal{I}\mathcal{I}}|} \\ &= \frac{|X^{\mathcal{I}\mathcal{I}} \cap Y^{\mathcal{I}\mathcal{I}}|}{|X^{\mathcal{I}\mathcal{I}}|} \\ &= \frac{|X'' \cap Y''|}{|X''|} \\ &= \frac{|(X' \cup Y')'|}{|X''|} \\ &= \mathrm{conf}_{\mathbb{K}_{\mathcal{I}}}(X' \to Y'). \end{aligned}$$

□

This already allows us to establish the first of Equation (5.5).

5.2.16 Corollary *Let $\mathcal{I}$ be a finite interpretation, and let $c \in [0,1]$. Then*

$$\mathrm{Conf}(\mathcal{I}, c) = \bigsqcap \mathrm{Conf}(\mathbb{K}_\mathcal{I}, c)$$

up to equivalence.

Proof Let $(X^\mathcal{I} \sqsubseteq Y^\mathcal{I}) \in \mathrm{Conf}(\mathcal{I}, c)$. Then $Y \subseteq X \subseteq \Delta^\mathcal{I}$ and $1 > \mathrm{conf}_\mathcal{I}(X^\mathcal{I} \sqsubseteq Y^\mathcal{I}) \geqslant c$. Thus, by Proposition 5.2.15, $1 > \mathrm{conf}_{\mathbb{K}_\mathcal{I}}(X' \to Y') \geqslant c$. Since $X' = X'''$, $Y' = Y'''$ and $X' \subseteq Y'$ we obtain that $(X' \to Y') \in \mathrm{Conf}(\mathbb{K}_\mathcal{I}, c)$. By Proposition 4.2.9, $\bigsqcap X' \equiv X^\mathcal{I}$ and $\bigsqcap Y' \equiv Y^\mathcal{I}$, thus $(X^\mathcal{I} \sqsubseteq Y^\mathcal{I}) \in \bigsqcap \mathrm{Conf}(\mathbb{K}_\mathcal{I}, c)$ up to equivalence.

Now let $(\bigsqcap A'' \sqsubseteq \bigsqcap B'') \in \bigsqcap \mathrm{Conf}(\mathbb{K}_\mathcal{I}, c)$. Then $A \subseteq B \subseteq M_\mathcal{I}$ and $1 > \mathrm{conf}_{\mathbb{K}_\mathcal{I}}(A'' \to B'') \geqslant c$. Define $X := A'$ and $Y := B'$. Then $1 > \mathrm{conf}_{\mathbb{K}_\mathcal{I}}(X' \to Y') \geqslant c$, and thus $1 > \mathrm{conf}_\mathcal{I}(X^\mathcal{I} \sqsubseteq Y^\mathcal{I}) \geqslant c$. Furthermore, $Y \subseteq X$, since $A \subseteq B$ implies $Y = B' \subseteq A' = X$. Again by Proposition 4.2.9 we have $\bigsqcap A'' = \bigsqcap X' \equiv X^\mathcal{I}$ and $\bigsqcap B'' = \bigsqcap Y' \equiv Y^\mathcal{I}$, and thus $(X^\mathcal{I} \sqsubseteq Y^\mathcal{I}) \in \mathrm{Conf}(\mathcal{I}, c)$ as required.

Let $A, \overline{A} \subseteq M_\mathcal{I}$. Then

$$\begin{aligned}
\bigsqcap A'' \equiv \bigsqcap \overline{A}'' &\implies (\bigsqcap A'')^\mathcal{I} = (\bigsqcap \overline{A}'')^\mathcal{I} \\
&\implies (A')^{\mathcal{I}\mathcal{I}} = (\overline{A}')^{\mathcal{I}\mathcal{I}} \\
&\implies A''' = \overline{A}''' \\
&\implies A'' = \overline{A}'',
\end{aligned}$$

using Proposition 4.2.9 and Proposition 4.2.10. Therefore, no two GCIs in $\bigsqcap \mathrm{Conf}(\mathbb{K}_\mathcal{I}, c)$ are equivalent, and the claim follows. □

To show the second equation of Equation (5.5) we proceed with another technical result.

5.2.17 Proposition *Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be a finite interpretation, and let $X, Y \subseteq \Delta^\mathcal{I}$. Then*

i. $X^\mathcal{I} \subsetneq Y^\mathcal{I}$ *implies* $\mathrm{pr}_{M_\mathcal{I}}(X^\mathcal{I}) \supsetneq \mathrm{pr}_{M_\mathcal{I}}(Y^\mathcal{I})$, *and*

ii. $X' \supsetneq Y'$ *implies* $\bigsqcap X' \subsetneq \bigsqcap Y'$

where the derivations are done in $\mathbb{K}_\mathcal{I}$.

Proof We already know from Corollary 4.2.11 that $X^\mathcal{I} \subsetneq Y^\mathcal{I}$ implies $\mathrm{pr}_{M_\mathcal{I}}(X^\mathcal{I}) \supseteq \mathrm{pr}_{M_\mathcal{I}}(Y^\mathcal{I})$ and that $X' \supsetneq Y'$ implies $\bigsqcap X' \sqsubseteq \bigsqcap Y'$.

Let us assume that $\mathrm{pr}_{M_\mathcal{I}}(X^\mathcal{I}) = \mathrm{pr}_{M_\mathcal{I}}(Y^\mathcal{I})$. Then by Lemma 4.3.5 and Lemma 4.2.7

$$X^\mathcal{I} \equiv \bigsqcap \mathrm{pr}_{M_\mathcal{I}}(X^\mathcal{I}) = \bigsqcap \mathrm{pr}_{M_\mathcal{I}}(Y^\mathcal{I}) \equiv Y^\mathcal{I}.$$

Conversely, if $\bigsqcap X' \equiv \bigsqcap Y'$, then using Corollary 4.2.11 we obtain

$$X' = \mathrm{pr}_{M_\mathcal{I}}(\bigsqcap X') = \mathrm{pr}_{M_\mathcal{I}}(\bigsqcap Y') = Y'$$

as required. □

5.2.18 Corollary *Let $\mathcal{I}$ be a finite interpretation, and let $c \in [0,1]$. Then*

$$\mathrm{Lux}(\mathcal{I}, c) = \bigsqcap \mathrm{Lux}(\mathbb{K}_\mathcal{I}, c)$$

up to equivalence.

Proof Using the same argumentation as in the proof of Corollary 5.2.16 it suffices to show that for $Y \subseteq Z \subseteq X \subseteq \Delta^\mathcal{I}$ it is true that

$$Y^\mathcal{I} \not\equiv Z^\mathcal{I} \not\equiv X^\mathcal{I} \iff Y' \neq Z' \neq X'. \tag{5.6}$$

Suppose first that $Y^\mathcal{I} \not\equiv Z^\mathcal{I} \not\equiv X^\mathcal{I}$. Then $Y^\mathcal{I} \subsetneq Z^\mathcal{I} \subsetneq X^\mathcal{I}$, and from Proposition 5.2.17 we obtain $\mathrm{pr}_{M_\mathcal{I}}(Y^\mathcal{I}) \supsetneq \mathrm{pr}_{M_\mathcal{I}}(Z^\mathcal{I}) \supsetneq \mathrm{pr}_{M_\mathcal{I}}(X^\mathcal{I})$. Proposition 4.2.8 then yields $Y' \supsetneq Z' \supsetneq X'$ as required.

Conversely, suppose $Y' \supsetneq Z' \supsetneq X'$. Then using Proposition 5.2.17 we obtain $\bigsqcap Y' \subsetneq \bigsqcap Z' \subsetneq \bigsqcap X'$, i. e. $Y^\mathcal{I} \subsetneq Z^\mathcal{I} \subsetneq X^\mathcal{I}$ by Proposition 4.2.9. □

5.2.4. Bases of Confident GCIs from Bases of Confident Implications

In the previous section we have obtained some first finite confident bases of $\mathrm{Th}_c(\mathcal{I})$ by mimicking the argumentation of Luxenburger's results in the setting of description logics. In this section we want to take another approach to obtain finite bases and finite confident bases, by directly transferring such bases of implications with high confidence to corresponding bases of GCIs with high confidence. More precisely, given a finite interpretation $\mathcal{I}$ and $c \in [0,1]$, we consider the induced formal context $\mathbb{K}_\mathcal{I}$ of $\mathcal{I}$, compute a base $\mathcal{L}$ of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$ and transfer this base into a base for $\mathrm{Th}_c(\mathbb{K})$ by defining

$$\bigsqcap \mathcal{L} := \{ \bigsqcap X \sqsubseteq \bigsqcap Y \mid (X \to Y) \in \mathcal{L} \}.$$

This approach has a particular advantage over computing finite bases of $\mathrm{Th}_c(\mathcal{I})$ the way we described it in the previous section. This advantage lies in the very close connection between formal concept analysis and data-mining, which extends to the level that formal concept analysis can be used as a framework for the logical foundations of the theory of *association rules* [105]. Although we have not introduced association rules formally here, they can be thought of as a generalization of implications with high confidence. Association rules are a well investigated topic in data-mining, and there is extensive literature on algorithms that mine association rules from data, see [58] for an overview over some of them. Because of the close connection of formal concept analysis to data-mining, these algorithms can be adapted quite easily to the problem of finding bases of implications with high confidence.

If we now establish another link between bases of implications with high confidence on the one hand, and bases of GCIs with high confidence on the other, then we can exploit the algorithms from data-mining for extracting association rules from data to find bases for GCIs with high confidence in finite interpretations. This link may be of particular interest

since these algorithms are usually tailored towards practical applications. We shall not go into details here, see Chapter 8.

The results presented in this section have been published before in [32].

We start with an observation that connects entailment between implications and entailment between GCIs.

5.2.19 Lemma *Let M be a set of concept descriptions over N_C and N_R, and let $\mathcal{L} \subseteq \mathrm{Imp}(M)$. Then for all $(X \to Y) \in \mathrm{Imp}(M)$ it is true that if $\mathcal{L} \models (X \to Y)$, then $\bigsqcap \mathcal{L} \models (\bigsqcap X \sqsubseteq \bigsqcap Y)$.*

Proof Let $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ be an interpretation over N_C and N_R such that $\mathcal{J} \models \bigsqcap \mathcal{L}$. Recall that we denote with $\mathbb{K}_{\mathcal{J},M}$ the induced context of $\mathcal{J}$ and M.

We shall first show that $\mathbb{K}_{\mathcal{J},M} \models \mathcal{L}$. To this end, let $(E \to F) \in \mathcal{L}$. Then $(\bigsqcap E)^{\mathcal{J}} \sqsubseteq (\bigsqcap F)^{\mathcal{J}}$, because $\mathcal{J} \models \bigsqcap \mathcal{L}$. By Proposition 4.2.9, $(\bigsqcap E)^{\mathcal{J}} = E'$ and $(\bigsqcap F)^{\mathcal{J}} = F'$, where the derivations are done in $\mathbb{K}_{\mathcal{J},M}$. Thus $E' \subseteq F'$, and $\mathbb{K}_{\mathcal{J},M} \models (E \to F)$. Hence, $\mathbb{K}_{\mathcal{J},M} \models \mathcal{L}$.

Since $\mathcal{L} \models (X \to Y)$, $\mathbb{K}_{\mathcal{J},M} \models (X \to Y)$, i. e. $X' \subseteq Y'$. By the same argument as before we obtain $(\bigsqcap X)^{\mathcal{J}} \subseteq (\bigsqcap Y)^{\mathcal{J}}$, and therefore $\mathcal{J} \models (\bigsqcap X \sqsubseteq \bigsqcap Y)$.

Since $\mathcal{J}$ was chosen arbitrarily, we obtain $\bigsqcap \mathcal{L} \models (\bigsqcap X \sqsubseteq \bigsqcap Y)$ as required. ▫

Note that we cannot expect the converse direction to hold in general as well. The reason for this is that entailment between implications does not "look inside" the attributes in the implications, but entailment between GCIs is allowed to consider the structure of concept descriptions. This is illustrated by the following example.

5.2.20 Example Let $N_C := \{ \mathsf{A}, \mathsf{B} \}$, $N_R := \{ \mathsf{r} \}$ and $M := \{ \mathsf{A}, \mathsf{B}, \exists \mathsf{r}.\mathsf{A}, \exists \mathsf{r}.\mathsf{B} \}$. Consider

$$\mathcal{L} := \{ \{ \mathsf{A} \} \to \{ \mathsf{B} \} \},$$
$$X := \{ \exists \mathsf{r}.\mathsf{A} \},$$
$$Y := \{ \exists \mathsf{r}.\mathsf{B} \}.$$

Then clearly $\mathcal{L} \not\models (X \to Y)$, but $\bigsqcap \mathcal{L} \models (\bigsqcap X \sqsubseteq \bigsqcap Y)$. ♢

The following proposition connects the notions of confidence of implications in finite formal contexts and confidence of GCIs in finite interpretations.

5.2.21 Proposition *Let $\mathcal{I}$ be a finite interpretation over N_C and N_R, let M be a set of concept descriptions over N_C and N_R and let $(X \to Y) \in \mathrm{Imp}(M)$. Then*

$$\mathrm{conf}_{\mathbb{K}_{\mathcal{I}}}(X \to Y) = \mathrm{conf}_{\mathcal{I}}(\textstyle\bigsqcap X \sqsubseteq \bigsqcap Y).$$

Proof In the following, all derivations are done in $\mathbb{K}_{\mathcal{I}}$.

By Proposition 4.2.9 we know that $X' = (\bigsqcap X)^{\mathcal{I}}$, thus if $X' = \emptyset$, then

$$\mathrm{conf}_{\mathbb{K}_{\mathcal{I}}}(X \to Y) = 1 = \mathrm{conf}_{\mathcal{I}}(\textstyle\bigsqcap X \sqsubseteq \bigsqcap Y).$$

Now suppose that $X' \neq \varnothing$. Then $(\bigsqcap X)^{\mathcal{I}} \neq \varnothing$, and we can compute

$$\begin{aligned}
\mathrm{conf}_{\mathbb{K}_{\mathcal{I}}}(X \to Y) &= \frac{|(X \cup Y)'|}{|X'|} \\
&= \frac{|X' \cap Y'|}{|X'|} \\
&= \frac{|(\bigsqcap X)^{\mathcal{I}} \cap (\bigsqcap Y)^{\mathcal{I}}|}{|(\bigsqcap X)^{\mathcal{I}}|} \\
&= \frac{|(\bigsqcap X \sqcap \bigsqcap Y)^{\mathcal{I}}|}{|(\bigsqcap X)^{\mathcal{I}}|} \\
&= \mathrm{conf}_{\mathcal{I}}(\bigsqcap X \sqsubseteq \bigsqcap Y)
\end{aligned}$$

again using Proposition 4.2.9. □

The main result of this section is now the following theorem.

5.2.22 Theorem *Let $\mathcal{I}$ be a finite interpretation over N_C and N_R, and let $c \in [0,1]$. Let $\mathcal{L}$ be a confident base of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$. Then $\bigsqcap \mathcal{L}$ is a confident base of $\mathrm{Th}_c(\mathcal{I})$.*

Proof We need to show that $\bigsqcap \mathcal{L} \subseteq \mathrm{Th}_c(\mathcal{I})$ and that $\bigsqcap \mathcal{L}$ is complete for $\mathrm{Th}_c(\mathcal{I})$.

To see that $\bigsqcap \mathcal{L}$ is sound for $\mathrm{Th}_c(\mathcal{I})$ let $(\bigsqcap X \sqsubseteq \bigsqcap Y) \in \bigsqcap \mathcal{L}$. Then $(X \to Y) \in \mathcal{L}$, and thus

$$\mathrm{conf}_{\mathcal{I}}(\bigsqcap X \sqsubseteq \bigsqcap Y) = \mathrm{conf}_{\mathbb{K}_{\mathcal{I}}}(X \to Y) \geqslant c$$

by Proposition 5.2.21. Thus $(\bigsqcap X \sqsubseteq \bigsqcap Y) \in \mathrm{Th}_c(\mathcal{I})$ and hence $\bigsqcap \mathcal{L} \subseteq \mathrm{Th}_c(\mathcal{I})$ as required.

For the completeness of $\bigsqcap \mathcal{L}$ for $\mathrm{Th}_c(\mathcal{I})$ we show two subclaims, namely

i. $\bigsqcap \mathcal{L} \models (\bigsqcap U \sqsubseteq (\bigsqcap U)^{\mathcal{I}\mathcal{I}})$ for all $U \subseteq M_{\mathcal{I}}$, and

ii. $\bigsqcap \mathcal{L} \models (X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}})$ for each $(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \in \mathrm{Conf}(\mathcal{I}, c)$.

The first claim then ensures that $\bigsqcap \mathcal{L}$ entails all GCIs from the set

$$\{\bigsqcap U \sqsubseteq (\bigsqcap U)^{\mathcal{I}\mathcal{I}} \mid U \subseteq M_{\mathcal{I}}\}$$

which by Theorem 4.3.6 is a base of $\mathcal{I}$. Showing (i) entails that $\bigsqcap \mathcal{L}$ is complete for $\mathrm{Th}(\mathcal{I})$. The claim (ii) states that $\bigsqcap \mathcal{L}$ is complete for $\mathrm{Conf}(\mathcal{I}, c)$. Using Corollary 5.2.14 and the fact that $\bigsqcap \mathcal{L}$ is sound for $\mathrm{Th}_c(\mathcal{I})$ then shows that $\bigsqcap \mathcal{L}$ is a confident base of $\mathrm{Th}_c(\mathcal{I})$.

We show (i). Let $U \subseteq M_{\mathcal{I}}$. Since $\mathcal{L}$ is a confident base for $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$ it is complete for $\mathbb{K}_{\mathcal{I}}$. Thus

$$\mathcal{L} \models (U \to U'').$$

Then Lemma 5.2.19 yields

$$\bigcap \mathcal{L} \models (\bigcap U \sqsubseteq \bigcap (U'')),$$

and by Lemma 4.2.13 we obtain

$$\bigcap \mathcal{L} \models (\bigcap U \sqsubseteq (\bigcap U)^{\mathcal{I}\mathcal{I}})$$

as required.

For (ii) let $(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \in \mathrm{Conf}(\mathcal{I}, c)$. Then $X, Y \subseteq \Delta^{\mathcal{I}}$ and $1 > \mathrm{conf}_{\mathcal{I}}(X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \geqslant c$. Since $X^{\mathcal{I}}$ and $Y^{\mathcal{I}}$ are expressible in terms of $M_{\mathcal{I}}$ by Lemma 4.3.5, we obtain from Proposition 4.2.9 that $X^{\mathcal{I}} \equiv \bigcap X'$ and $Y^{\mathcal{I}} \equiv \bigcap Y'$. Thus,

$$\bigcap \mathcal{L} \models (X^{\mathcal{I}} \sqsubseteq Y^{\mathcal{I}}) \iff \bigcap \mathcal{L} \models (\bigcap X' \sqsubseteq \bigcap Y'). \tag{5.7}$$

By Proposition 5.2.21

$$\mathrm{conf}_{\mathbb{K}_{\mathcal{I}}}(X' \to Y') = \mathrm{conf}_{\mathcal{I}}(\bigcap X' \sqsubseteq \bigcap Y') \geqslant c.$$

Since $\mathcal{L}$ is a base of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$ we obtain $\mathcal{L} \models (X' \to Y')$, thus by Lemma 5.2.19

$$\bigcap \mathcal{L} \models (\bigcap X' \sqsubseteq \bigcap Y')$$

which together with Equation (5.7) yields the claim. □

Since $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$ is a confident base of itself, the theorem immediately yields that $\bigcap \mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$ is a finite confident base of $\mathrm{Th}_c(\mathcal{I})$.

Theorem 5.2.22 has the drawback that $\bigcap \mathcal{L}$ still contains trivial knowledge in the sense that whenever $C, D \in M_{\mathcal{I}}$ are such that $C \sqsubseteq D$, then $\mathcal{L}$ has to entail $\{ C \} \to \{ D \}$, although the corresponding GCI $C \sqsubseteq D$ is trivially true. We can remedy this redundancy by considering an appropriate background knowledge. Recall that in Equation (4.8) we defined the set

$$\mathcal{S}_{\mathcal{I}} := \{ \{ C \} \to \{ D \} \mid C, D \in M_{\mathcal{I}}, C \sqsubseteq D \}.$$

5.2.23 Corollary *Let $\mathcal{I}$ be a finite interpretation over N_C and N_R, let $c \in [0,1]$ and $\mathcal{L} \subseteq \mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$ be such that $\mathcal{L} \cup \mathcal{S}_{\mathcal{I}}$ is a confident base of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$. Then $\bigcap \mathcal{L}$ is a finite confident base of $\mathrm{Th}_c(\mathcal{I})$.*

Proof By Theorem 5.2.22 the set $\bigcap \mathcal{L} \cup \bigcap \mathcal{S}_{\mathcal{I}}$ is a finite confident base of $\mathrm{Th}_c(\mathcal{I})$. Since $\bigcap \mathcal{S}_{\mathcal{I}}$ is valid in every interpretation, it is entailed by $\bigcap \mathcal{L}$, and thus the set $\bigcap \mathcal{L}$ is already complete for $\mathrm{Th}_c(\mathcal{I})$. Thus, $\bigcap \mathcal{L}$ is a finite confident base of $\mathrm{Th}_c(\mathcal{I})$. □

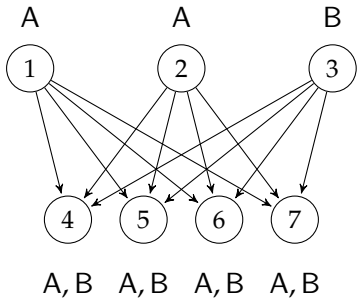


Figure 5.2.: Example interpretation $\mathcal{I}$ for Example 5.2.24.

We can use Lemma 5.2.19 to further remove redundancies from bases obtained as in Theorem 5.2.22. More precisely, if $\mathcal{L}$ is a base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$, and if $(X \to Y) \in \mathcal{L}$ is such that

$$\mathcal{L} \backslash \{ X \to Y \} \models (X \to Y),$$

then Lemma 5.2.19 yields that

$$\bigsqcap \mathcal{L} \backslash \{ \bigsqcap X \sqsubseteq \bigsqcap Y \} \models (\bigsqcap X \sqsubseteq \bigsqcap Y).$$

Thus, redundancies in bases of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$ always result in redundancies in bases of $\mathrm{Th}_c(\mathcal{I})$. Thus, removing these redundancies is a good starting point to obtain smaller bases of $\mathrm{Th}_c(\mathcal{I})$.

In particular, we can consider irredundant bases $\mathcal{L}$ of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$. However, even if $\mathcal{L}$ is irredundant, $\bigsqcap \mathcal{L}$ may contain redundancies, as the following example shows.

5.2.24 Example We are looking for a finite interpretation $\mathcal{I}$, a $c \in [0, 1]$, and a non-redundant set $\mathcal{L}$ such that $\mathcal{L} \cup \mathcal{S}_\mathcal{I}$ is a confident base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$, but $\bigsqcap \mathcal{L}$ contains redundancies.

For this we employ a similar idea as we did in Example 5.2.20. More precisely, we want to construct an interpretation $\mathcal{I}$ such that for two concept names $\mathsf{A}, \mathsf{B} \in N_C$ and $\mathsf{r} \in N_R$, both implications $\{ \mathsf{A} \} \to \{ \mathsf{B} \}$ and $\{ \exists \mathsf{r}.\mathsf{A} \} \to \{ \exists \mathsf{r}.(\mathsf{A} \sqcap \mathsf{B}) \}$ have confidence at least c in $\mathbb{K}_\mathcal{I}$. Then by Proposition 5.2.21, $\mathsf{A} \sqsubseteq \mathsf{B}$ and $\exists \mathsf{r}.\mathsf{A} \sqsubseteq \exists \mathsf{r}.(\mathsf{A} \sqcap \mathsf{B})$ have confidence at least c in $\mathcal{I}$. If we can include these two implications in an irredundant base $\mathcal{L}$ of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$ with background knowledge $\mathcal{S}_\mathcal{I}$, then $\bigsqcap \mathcal{L}$ contains redundancies, as desired.

So let $N_C := \{ \mathsf{A}, \mathsf{B} \}$ and $N_R := \{ \mathsf{r} \}$. Consider the interpretation $\mathcal{I}$ as given in Figure 5.2, where every edge is labeled with r. Then

$$\begin{aligned} \mathrm{conf}_\mathcal{I}(\mathsf{A} \sqsubseteq \mathsf{B}) &= \frac{2}{3} \\ \mathrm{conf}_\mathcal{I}(\exists \mathsf{r}.\mathsf{A} \sqsubseteq \exists \mathsf{r}.\mathsf{B}) &= 1, \end{aligned}$$

$\mathbb{K}_\mathcal{I}$	$\bot$	A	B	$\exists r.(A \sqcap \exists r.(A \sqcap B))$	$\exists r.(B \sqcap \exists r.(A \sqcap B))$	$\exists r.(A \sqcap B)$	$\exists r.\exists r.(A \sqcap B)$	$\exists r.A$	$\exists r.B$	$\exists r.\top$
1		×				×		×	×	×
2		×				×		×	×	×
3			×			×		×	×	×
4		×	×							
5		×	×							
6		×	×							
7		×	×							

Figure 5.3.: Induced formal context of $\mathcal{I}$ as in Example 5.2.24.

thus following our argumentation from above we can choose $c = \frac{1}{2}$, say. Then we want to find a irredundant set $\mathcal{L}$ of implications such that $\mathcal{L} \cup \mathcal{S}_\mathcal{I}$ is a confident base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$ and $\mathcal{L}$ contains both $\{ A \} \to \{ B \}$ and $\{ \exists r.A \} \to \{ \exists r.(A \sqcap B) \}$. We find

$$M_\mathcal{I} = \{ \bot, A, B, \exists r.(A \sqcap \exists r.(A \sqcap B)), \\ \exists r.(B \sqcap \exists r.(A \sqcap B)), \exists r.(A \sqcap B), \exists r.\exists r.(A \sqcap B), \exists r.A, \exists r.B, \exists r.\top \},$$

and $\mathbb{K}_\mathcal{I}$ is as shown in Figure 5.3.
By Theorem 5.2.11 the set $\mathrm{Can}(\mathbb{K}_\mathcal{I}) \cup \mathrm{Conf}(\mathcal{I}, c)$ is a finite confident base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$. An irredundant subset $\mathcal{L}$ of this base which is still a base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$ with background knowledge $\mathcal{S}_\mathcal{I}$ is given by the following list of implications:

$$\begin{aligned}
\varnothing &\to \{ A \}, \\
\{ A \} &\to \{ B \}, \\
\{ \exists r.A \} &\to \{ \exists r.(A \sqcap B) \}, \\
\{ \exists r.B \} &\to \{ \exists r.(A \sqcap B) \}, \\
\{ \exists r.\top \} &\to \{ \exists r.(A \sqcap B) \}, \\
\{ A, B, \exists r.(A \sqcap B) \} &\to \{ \bot \}, \\
\{ \exists r.\exists r.(A \sqcap B) \} &\to \{ \bot \}, \\
\{ \exists r.(B \sqcap \exists r.(A \sqcap B)) \} &\to \{ \bot \}, \\
\{ \exists r.(A \sqcap \exists r.(A \sqcap B)) \} &\to \{ \bot \}.
\end{aligned}$$

Then $\mathcal{L}$ is irredundant and by Corollary 5.2.23 the set $\bigcap \mathcal{L}$ is a base of $\mathrm{Th}_c(\mathcal{I})$. But $\bigcap \mathcal{L}$ contains the GCIs $A \sqsubseteq B$ and $\exists r.A \sqsubseteq \exists r.(A \sqcap B)$, and thus $\bigcap \mathcal{L}$ is not irredundant. ♢

Instead of only considering irredundant bases of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$, we can go even a step further and consider *minimal* bases of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$. For this we recall that if $\mathcal{L}$ is a base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$, then the canonical base $\mathrm{Can}(\mathcal{L})$ of $\mathcal{L}$ is a minimal base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$.[8] Since $\mathrm{Cn}(\mathcal{L}) = \mathrm{Cn}(\mathrm{Can}(\mathcal{L}))$, we know that $\mathrm{Can}(\mathcal{L})$ is also a base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$. In particular, if $\mathcal{L} = \mathrm{Th}_c(\mathbb{K}_\mathcal{I})$ we can consider $\mathrm{Can}(\mathrm{Th}_c(\mathbb{K}_\mathcal{I}))$ as a minimal base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$.[9] In this case we also obtain that $\prod \mathrm{Can}(\mathrm{Th}_c(\mathbb{K}_\mathcal{I}))$ is a finite base of $\mathrm{Th}_c(\mathcal{I})$, as the following argumentation shows.

5.2.25 Corollary *Let $\mathcal{I}$ be a finite interpretation, let $c \in [0,1]$ and let $\mathcal{K} \subseteq \mathrm{Imp}(M_\mathcal{I})$ be a base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$. Then $\prod \mathcal{K}$ is a base of $\mathrm{Th}_c(\mathcal{I})$.*

Proof By Theorem 5.2.22 we know that $\prod \mathrm{Th}_c(\mathbb{K}_\mathcal{I})$ is a finite confident base of $\mathrm{Th}_c(\mathcal{I})$. As $\mathcal{K}$ is a base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$ we can infer from Lemma 5.2.19 that $\prod \mathcal{K}$ is also complete for $\prod \mathrm{Th}_c(\mathbb{K}_\mathcal{I})$. We thus obtain that $\prod \mathcal{K}$ is complete for $\mathrm{Th}_c(\mathcal{I})$.

On the other hand, since $\mathcal{K}$ is a base of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$, it is also true that $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$ is a base of $\mathcal{K}$. But then all implications in $\mathcal{K}$ are entailed by $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$, and Lemma 5.2.19 yields that all GCIs in $\prod \mathcal{K}$ are entailed by $\prod \mathrm{Th}_c(\mathbb{K}_\mathcal{I}) \subseteq \mathrm{Th}_c(\mathcal{I})$. Thus, $\prod \mathcal{K}$ is also sound for $\mathrm{Th}_c(\mathcal{I})$, and in sum we obtain that $\prod \mathcal{K}$ is a finite base of $\mathrm{Th}_c(\mathcal{I})$. □

The approach of considering the canonical base of $\mathrm{Th}_c(\mathbb{K})$ has the potential drawback, however, that we cannot guarantee anymore that the base itself is a confident base, i. e. it can happen that $\prod \mathrm{Can}(\mathrm{Th}_c(\mathbb{K})) \subseteq \mathrm{Th}_c(\mathcal{I})$ does not hold.

5.2.5. Completing Sets of GCIs

The bases we have obtained in Corollary 5.2.14 consisted of two parts, namely a complete subset $\mathcal{B}$ of $\mathrm{Lux}(\mathcal{I}, c)$ and a set $\mathcal{L}$ of valid GCIs such that $\mathcal{L} \cup \mathcal{B}$ is complete for $\mathcal{I}$. In this section we are going to show that we can, given the set $\mathcal{B}$, compute the set $\mathcal{L}$ in such a way that $\mathcal{L} \cup \mathcal{B}$ is complete for $\mathrm{Th}(\mathcal{I})$.

The results of this section have previously been published in [28].

The idea we want to exploit for this is borrowed from formal concept analysis: if $\mathcal{B}$ is a set of implications, then we can find a set $\mathcal{L}$ of implications valid in a formal context $\mathbb{K}$ such that $\mathcal{L}$ has minimal cardinality. More precisely, the set

$$\mathcal{L} := \mathrm{Can}(\mathbb{K}, \mathcal{B})$$

has this property by Theorem 2.4.7. What we want to do in this section is to lift this result to the level of general concept inclusions.

[8] Note that we have only introduced the canonical base for formal contexts, but by Proposition 2.3.7 we can represent every set $\mathcal{L}$ as a base of a formal context $\mathbb{K}_\mathcal{L}$. Then $\mathrm{Can}(\mathcal{L}) := \mathrm{Can}(\mathbb{K}_\mathcal{L})$. Note that this definition is independent from the context $\mathbb{K}_\mathcal{L}$ we use.

[9] Also note that $\mathrm{Can}(\mathcal{L}) = \mathrm{Can}(\mathrm{Th}_c(\mathbb{K}_\mathcal{I}))$ for all bases $\mathcal{L}$ of $\mathrm{Th}_c(\mathbb{K}_\mathcal{I})$.

To this end, we need to transform sets of general concept inclusions into sets of implications. For this, we make use of projections pr_M as introduced in Section 4.2.2. More precisely, if M is a set of concept descriptions and $\mathcal{B}$ is a set of GCIs, then we define

$$\mathrm{pr}_M(\mathcal{B}) := \{\, \mathrm{pr}_M(C) \to \mathrm{pr}_M(D) \mid (C \sqsubseteq D) \in \mathcal{B} \,\}.$$

For $M = M_{\mathcal{I}}$, we can take the set $\mathrm{pr}_{M_{\mathcal{I}}}(\mathcal{B})$ and compute $\mathrm{Can}(\mathbb{K}_{\mathcal{I}}, \mathrm{pr}_{M_{\mathcal{I}}}(\mathcal{B}))$. It is then true that

$$\mathcal{B} \cup \{\, \textstyle\bigcap U \sqsubseteq (\bigcap U)^{\mathcal{I}\mathcal{I}} \mid (U \to U'') \in \mathrm{Can}(\mathbb{K}_{\mathcal{I}}, \mathrm{pr}_{M_{\mathcal{I}}}(\mathcal{B})) \,\}$$

is complete for $\mathcal{I}$, provided that the concept descriptions in $\mathcal{B}$ are all expressible in terms of $M_{\mathcal{I}}$.

This result already appeared in [41, Theorem 5.12], however only for the case that $\mathcal{B}$ is empty. We shall generalize this result to also cover the case that $\mathcal{B}$ contains arbitrary GCIs. The proof of this generalization is similar to the one of [41, Theorem 5.12].

5.2.26 Theorem *Let $\mathcal{I}$ be a finite interpretation over N_C and N_R, and let $\mathcal{B}$ be a set of GCIs over N_C and N_R, where all concept descriptions appearing in $\mathcal{B}$ are expressible in terms of $M_{\mathcal{I}}$. Let $\mathcal{L} \subseteq \mathrm{Th}(\mathbb{K}_{\mathcal{I}})$ such that*

i. $\mathcal{L} \cup \mathrm{pr}_{M_{\mathcal{I}}}(\mathcal{B})$ is complete for $\mathbb{K}_{\mathcal{I}}$, and

ii. $\mathcal{L}$ only contains implications of the form $A \to A''$ with $A \subseteq M_{\mathcal{I}}$.

Then $\bigcap \mathcal{L} \cup \mathcal{B}$ is complete for $\mathcal{I}$.

Proof We show that for each $U \subseteq M_{\mathcal{I}}$ it is true that

$$\textstyle\bigcap \mathcal{L} \cup \mathcal{B} \models (\bigcap U \sqsubseteq (\bigcap U)^{\mathcal{I}\mathcal{I}}).$$

If we establish this fact, then Theorem 4.3.6 immediately yields that $\bigcap \mathcal{L} \cup \mathcal{B}$ is complete for $\mathrm{Th}(\mathcal{I})$.

Let $\mathcal{J}$ be a finite interpretation such that $\mathcal{J} \models (\bigcap \mathcal{L} \cup \mathcal{B})$. Let us write $\cdot'^{\mathcal{I}}$ for the derivation operators in $\mathbb{K}_{\mathcal{I},M_{\mathcal{I}}}$, and $\cdot'^{\mathcal{J}}$ for the derivation operators in $\mathbb{K}_{\mathcal{J},M_{\mathcal{I}}}$. We shall then show the following claims

i. $\mathbb{K}_{\mathcal{J},M_{\mathcal{I}}} \models (\mathcal{L} \cup \mathrm{pr}_{M_{\mathcal{I}}}(\mathcal{B}))$,

ii. $\mathbb{K}_{\mathcal{J},M_{\mathcal{I}}} \models (U \to U'^{\mathcal{I}\mathcal{I}})$ for all $U \subseteq M_{\mathcal{I}}$, and finally

iii. $\mathcal{J} \models (\bigcap U \sqsubseteq (\bigcap U)^{\mathcal{I}\mathcal{I}})$ for all $U \subseteq M_{\mathcal{I}}$.

To show the first claim we start with some preparations. Let $U \subseteq M_{\mathcal{I}}$. Then by Proposition 4.2.9 it is true that

$$(\bigsqcap U)^{\mathcal{J}} = U'^{\mathcal{J}}. \tag{5.8}$$

Furthermore, the concept description $(\bigsqcap U)^{\mathcal{I}\mathcal{I}}$ is expressible in terms of $M_{\mathcal{I}}$ by Lemma 4.3.5. From this we can infer with Lemma 4.2.7 that

$$(\bigsqcap U)^{\mathcal{I}\mathcal{I}} \equiv \bigsqcap \mathrm{pr}_{M_{\mathcal{I}}}((\bigsqcap U)^{\mathcal{I}\mathcal{I}}).$$

Then Corollary 4.2.11 yields

$$\begin{aligned}((\bigsqcap U)^{\mathcal{I}\mathcal{I}})^{\mathcal{J}} &= (\bigsqcap \mathrm{pr}_{M_{\mathcal{I}}}((\bigsqcap U)^{\mathcal{I}\mathcal{I}}))^{\mathcal{J}} \\ &= (\mathrm{pr}_{M_{\mathcal{I}}}((\bigsqcap U)^{\mathcal{I}\mathcal{I}}))'^{\mathcal{J}} \\ &= U'^{\mathcal{I}'\mathcal{I}'\mathcal{J}}.\end{aligned} \tag{5.9}$$

Now let $(U \to U'^{\mathcal{I}'\mathcal{I}}) \in \mathcal{L}$. Then $\mathcal{J} \models (\bigsqcap U \sqsubseteq (\bigsqcap U)^{\mathcal{I}\mathcal{I}})$, and therefore

$$(\bigsqcap U)^{\mathcal{J}} \subseteq ((\bigsqcap U)^{\mathcal{I}\mathcal{I}})^{\mathcal{J}},$$

and Equation (5.8) and Equation (5.9) yield

$$U'^{\mathcal{J}} \subseteq U'^{\mathcal{I}'\mathcal{I}'\mathcal{I}'\mathcal{J}},$$

i. e. $\mathbb{K}_{\mathcal{J},M_{\mathcal{I}}} \models (U \to U'^{\mathcal{I}'\mathcal{I}})$. Thus, $\mathbb{K}_{\mathcal{J},M_{\mathcal{I}}} \models \mathcal{L}$.

Let $(C \sqsubseteq D) \in \mathcal{B}$. It remains to show that $\mathrm{pr}_{M_{\mathcal{I}}}(C) \to \mathrm{pr}_{M_{\mathcal{I}}}(D)$ holds in $\mathbb{K}_{\mathcal{J},M_{\mathcal{I}}}$. It is true that $\mathcal{J} \models (C \sqsubseteq D)$, i. e. $C^{\mathcal{J}} \subseteq D^{\mathcal{J}}$. Since both C, D are expressible in terms of $M_{\mathcal{I}}$, Lemma 4.2.7 yields

$$(\bigsqcap \mathrm{pr}_{M_{\mathcal{I}}}(C))^{\mathcal{J}} \subseteq (\bigsqcap \mathrm{pr}_{M_{\mathcal{I}}}(D))^{\mathcal{J}}$$

and thus, using Equation (5.8) again,

$$\mathrm{pr}_{M_{\mathcal{I}}}(C)'^{\mathcal{J}} \subseteq \mathrm{pr}_{M_{\mathcal{I}}}(D)'^{\mathcal{J}},$$

i. e. $\mathbb{K}_{\mathcal{J},M_{\mathcal{I}}} \models (\mathrm{pr}_{M_{\mathcal{I}}}(C) \to \mathrm{pr}_{M_{\mathcal{I}}}(D))$. Thus we have shown that $\mathbb{K}_{\mathcal{J},M_{\mathcal{I}}} \models \mathcal{B}$. This proves the first claim.

Now let $U \subseteq M_{\mathcal{I}}$. Then $\mathbb{K}_{\mathcal{I}} \models (U \to U'^{\mathcal{I}'\mathcal{I}})$. Since $\mathcal{L} \cup \mathrm{pr}_{M_{\mathcal{I}}}(\mathcal{B})$ is complete for $\mathbb{K}_{\mathcal{I}}$, we obtain

$$\mathcal{L} \cup \mathrm{pr}_{M_{\mathcal{I}}}(\mathcal{B}) \models (U \to U'^{\mathcal{I}'\mathcal{I}}).$$

Since $\mathbb{K}_{\mathcal{J},M_\mathcal{I}} \models \mathcal{L} \cup \mathrm{pr}_{M_\mathcal{I}}(\mathcal{B})$, it is true that

$$\mathbb{K}_{\mathcal{J},M_\mathcal{I}} \models (U \to U'^{\mathcal{I}'\mathcal{I}}),$$

i. e. $U'^{\mathcal{J}} \subseteq U'^{\mathcal{I}'\mathcal{I}'^{\mathcal{J}}}$. Using Equation (5.8) and Equation (5.9) again yields

$$(\bigsqcap U)^{\mathcal{J}} \subseteq ((\bigsqcap U)^{\mathcal{I}\mathcal{I}})^{\mathcal{J}},$$

i. e. $\mathcal{J} \models (\bigsqcap U \sqsubseteq (\bigsqcap U)^{\mathcal{I}\mathcal{I}})$. Since $U \subseteq M_\mathcal{I}$ was chosen arbitrarily, we have thus shown that $\bigsqcap \mathcal{L} \cup \mathcal{B}$ is complete for $\mathcal{I}$. □

We can apply this theorem to our setting of computing finite confident bases as follows. Let $\mathcal{C} \subseteq \mathrm{Lux}(\mathcal{I}, c)$ be complete for $\mathrm{Lux}(\mathcal{I}, c)$. If we compute $\mathcal{L} := \mathrm{Can}(\mathbb{K}_\mathcal{I}, \mathrm{pr}_{M_\mathcal{I}}(\mathcal{B}))$, then $\mathcal{L}$ is as required by the above theorem, and thus $\bigsqcap \mathcal{L} \cup \mathcal{B}$ is complete for $\mathcal{I}$. By Corollary 5.2.14 the set $\bigsqcap \mathcal{L} \cup \mathcal{C}$ is a finite confident base of $\mathrm{Th}_c(\mathcal{I})$.

Of course, we can also include the implications in $\mathcal{S}_\mathcal{I}$ as background knowledge when computing $\mathcal{L}$. To see this we observe that the proof of Theorem 5.2.26 still works if we include $\mathcal{S}_\mathcal{I}$ in the computation of $\mathcal{L}$. Part(i) of the proof would then be extended to also claim that $\mathbb{K}_{\mathcal{J},M_\mathcal{I}} \models \mathcal{S}_\mathcal{I}$, which is true for all induced contexts with attribute set $M_\mathcal{I}$.

5.2.27 Corollary *Let $\mathcal{I}$ be a finite interpretation over N_C and N_R, and let $\mathcal{C} \subseteq \mathrm{Lux}(\mathcal{I}, c)$ be complete for $\mathrm{Lux}(\mathcal{I}, c)$. Define*

$$\mathcal{L} := \mathrm{Can}(\mathbb{K}_\mathcal{I}, \mathrm{pr}_{M_\mathcal{I}}(\mathcal{C}) \cup \mathcal{S}_\mathcal{I}).$$

Then $\bigsqcap \mathcal{L} \cup \mathcal{C}$ is a finite confident base of $\mathrm{Th}_c(\mathcal{I})$.

For $c = 0$ we obtain $\mathrm{Lux}(\mathcal{I}, c) = \varnothing$, and thus we are back in the case of valid GCIs. For this we know from Theorem 4.3.8 that the set $\bigsqcap \mathcal{L}$ in the previous corollary is *minimal* with respect to being a base of $\mathcal{I}$. A natural question is now to ask whether we can also expect such a minimality result to be true in the case of GCIs with high confidence, i. e. whether $\bigsqcap \mathcal{L}$ is minimal with respect to $\bigsqcap \mathcal{L} \cup \mathcal{B}$ being a finite confident base of $\mathrm{Th}_c(\mathcal{I})$. The next example shows that this is not the case.

5.2.28 Example Let $N_C = \{ \mathsf{A}, \mathsf{B} \}$ and $N_R = \{ \mathsf{r} \}$ and consider the finite interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ as given in Figure 5.4.

If $X \subseteq \Delta^\mathcal{I}$ is such that $5 \in X$, then $X^\mathcal{I} = \mathsf{A}$. If $5 \notin X$ but $1 \in X$ or $4 \in X$, then $X^\mathcal{I} = \mathsf{A} \sqcap \mathsf{B}$. Otherwise, $X^\mathcal{I} = \mathsf{A} \sqcap \mathsf{B} \sqcap \exists \mathsf{r}.(\mathsf{A} \sqcap \mathsf{B})$. Thus the set of model-based most-specific concept descriptions of $\mathcal{I}$ is (up to equivalence)

$$\{ \bot, \mathsf{A}, \mathsf{A} \sqcap \mathsf{B}, \mathsf{A} \sqcap \mathsf{B} \sqcap \exists \mathsf{r}.(\mathsf{A} \sqcap \mathsf{B}) \}$$

and thus we obtain

$$M_\mathcal{I} := \{ \bot, \mathsf{A}, \mathsf{B}, \exists \mathsf{r}.\mathsf{A}, \exists \mathsf{r}.(\mathsf{A} \sqcap \mathsf{B}), \exists \mathsf{r}.(\mathsf{A} \sqcap \mathsf{B} \sqcap \exists \mathsf{r}.(\mathsf{A} \sqcap \mathsf{B})) \}.$$

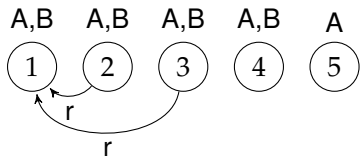


Figure 5.4.: Interpretation for Example 5.2.28

	$\bot$	A	B	$\exists$r.A	$\exists$r.(A $\sqcap$ B)	$\exists$r.(A $\sqcap$ B $\sqcap$ $\exists$r.(A $\sqcap$ B))
1		×	×			
2		×	×	×	×	
3		×	×	×	×	
4		×	×			
5		×				

Figure 5.5.: Induced Context of Figure 5.4

The induced context is shown in Figure 5.5.

Let $c = \frac{4}{5}$. Then $\mathrm{Conf}(\mathbb{K}_\mathcal{I}, c) = \{ \{ \mathsf{A} \} \to \{ \mathsf{A}, \mathsf{B} \} \}$, thus

$$\mathrm{Conf}(\mathcal{I}, c) = \{ \mathsf{A} \sqsubseteq \mathsf{A} \sqcap \mathsf{B} \}.$$

Set $\mathcal{B} := \mathrm{Conf}(\mathcal{I}, c)$. Then $\mathrm{pr}_{M_\mathcal{I}}(\mathcal{B}) = \mathrm{Conf}(\mathbb{K}_\mathcal{I}, c)$. Let

$$U := \{ \exists \mathsf{r}.\mathsf{A}, \mathsf{A}, \mathsf{B} \}.$$

Then U is closed under $\mathcal{S}_\mathcal{I}$. Furthermore, U is a $\mathrm{pr}_{M_\mathcal{I}}(\mathcal{B})$-pseudo-intent of $\mathbb{K}_\mathcal{I}$: $\varnothing$ is a $\mathrm{pr}_{M_\mathcal{I}}(\mathcal{B})$-pseudo-intent of $\mathbb{K}_\mathcal{I}$, and $\varnothing'' = \{ \mathsf{A} \} \subseteq U$. The set $\{ \mathsf{A} \}$ is an intent of $\mathbb{K}_\mathcal{I}$, and the sets $\{ \mathsf{B} \}$ and $\{ \exists \mathsf{r}.\mathsf{A} \}$ are not supersets of $\varnothing'' = \{ \mathsf{A} \}$. The set $\{ \mathsf{A}, \mathsf{B} \}$ is again an intent of $\mathbb{K}_\mathcal{I}$, $\{ \exists \mathsf{r}.\mathsf{A}, \mathsf{A} \}$ is not closed under $\mathrm{pr}_{M_\mathcal{I}}(\mathcal{B})$ and $\{ \exists \mathsf{r}.\mathsf{A}, \mathsf{B} \}$ does not contain A. Thus, the only $\mathrm{pr}_{M_\mathcal{I}}(\mathcal{B})$-pseudo-intent of $\mathbb{K}_\mathcal{I}$ contained in U is $\varnothing$, and its closure is again contained in U. Thus, U is a $\mathrm{pr}_{M_\mathcal{I}}(\mathcal{B})$-pseudo-intent of $\mathbb{K}_\mathcal{I}$.

Therefore, $(U \to U'') \in \mathrm{Can}(\mathbb{K}_\mathcal{I}, \mathrm{pr}_{M_\mathcal{I}}(\mathcal{B}) \cup \mathcal{S}_\mathcal{I})$, i. e.

$$(\{ \exists \mathsf{r}.\mathsf{A}, \mathsf{A}, \mathsf{B} \} \to \{ \exists \mathsf{r}.(\mathsf{A} \sqcap \mathsf{B}) \}) \in \mathrm{Can}(\mathbb{K}_\mathcal{I}, \mathrm{pr}_{M_\mathcal{I}}(\mathcal{B}) \cup \mathcal{S}_\mathcal{I}).$$

This yields that

$$(\exists \mathsf{r}.\mathsf{A} \sqcap \mathsf{A} \sqcap \mathsf{B} \sqsubseteq \exists \mathsf{r}.(\mathsf{A} \sqcap \mathsf{B})) \in \bigsqcap \mathrm{Can}(\mathbb{K}_\mathcal{I}, \mathrm{pr}_{M_\mathcal{I}}(\mathcal{B}) \cup \mathcal{S}_\mathcal{I}).$$

But this GCI is entailed by $\mathcal{B}$, so the set $\bigsqcap \mathrm{Can}(\mathbb{K}_\mathcal{I}, \mathrm{pr}_{M_\mathcal{I}}(\mathcal{B}) \cup \mathcal{S}_\mathcal{I}) \cup \mathcal{B}$ is not irredundant. In particular, $\bigsqcap \mathrm{Can}(\mathbb{K}_\mathcal{I}, \mathrm{pr}_{M_\mathcal{I}}(\mathcal{B}) \cup \mathcal{S}_\mathcal{I})$ is not minimal with respect to the property that it forms together with $\mathcal{B}$ a confident base of $\mathrm{Th}_c(\mathcal{I})$. ♢

The main reason why the minimality statement fails is that entailment between GCIs can happen "behind the quantifier": if $\mathcal{B} = \{ \mathsf{A} \sqsubseteq \mathsf{B} \}$, then $\mathcal{B}$ entails $\exists \mathsf{r}.\mathsf{A} \sqsubseteq \exists \mathsf{r}.\mathsf{B}$. This entailment process can usually not be simulated by implications, as they are not allowed to consider the structure of the attributes in a formal context.

It is possible to find a special case where entailment behind the quantifier cannot happen: if $\mathcal{B}$ is a set of *valid* GCIs of $\mathcal{I}$, and if $C = \bigsqcap U$ for $U \subseteq M_{\mathcal{I}}$ and $U \neq \varnothing$. In this case, C can be written as

$$C = \bigsqcap V \sqcap \bigsqcap_{(r,Z)\in\Pi} \exists r.Z^{\mathcal{I}}$$

for some $V \subseteq N_C$ and some $\Pi \subseteq N_R \times \mathfrak{P}(\Delta^{\mathcal{I}})$. Then one can argue that the concept descriptions $Z^{\mathcal{I}}$ behind the quantifier are already "closed under entailment" with respect to $\mathcal{B}$, because they are model-based most-specific concept descriptions and all GCIs in $\mathcal{B}$ are valid in $\mathcal{I}$. Thus, if a GCI $C \sqsubseteq D$ is entailed by $\mathcal{B}$, it must happen on the top-level of the concept descriptions, and this entailment can then be simulated by implications.

A formal argumentation requires the notions of *simulations* between $\mathcal{EL}$ description graphs, which have not been introduced in this work. For more details on this we refer to Lemma 5.16 and the proof of Theorem 5.18 in [41].

5.2.6. Unravelling $\mathcal{EL}^{\perp}_{\mathbf{gfp}}$ Bases into $\mathcal{EL}^{\perp}$ Bases

So far we have only considered finite bases of $\mathrm{Th}_c(\mathcal{I})$ which may contain proper $\mathcal{EL}^{\perp}_{\mathrm{gfp}}$ concept descriptions. As we had argued before, those concept descriptions may actually be hard to read, even for those trained in logics. Therefore, it would be desirable to obtain bases which contain $\mathcal{EL}^{\perp}$ concept descriptions only, as these are potentially much easier to understand. To show that this is indeed possible is the purpose of this section. The argumentation we want to employ is again similar to the one used by [41]; see Section 4.3.

As a first step, as already discussed in Section 4.3, we define an auxiliary set $\mathcal{X}_{\mathcal{I}}$ that only contains $\mathcal{EL}^{\perp}$ concept descriptions but "captures" entailment between $\mathcal{EL}^{\perp}_{\mathrm{gfp}}$ concept descriptions. For this, recall that Lemma 4.3.9 states that for a finite interpretation $\mathcal{I}$ and all $\mathcal{EL}^{\perp}_{\mathrm{gfp}}$ concept descriptions $C = (A, \mathcal{T})$ it is true that

$$C^{\mathcal{I}} = (C_d)^{\mathcal{I}},$$

where $d = |\Delta^{\mathcal{I}}| \cdot |N_D(\mathcal{T})| + 1$. Note that the constant d depends on the concept description C. To emphasize this dependency, we shall write d_C instead of just d.

The set $\mathcal{X}_{\mathcal{I}}$ is now defined as

$$\mathcal{X}_{\mathcal{I}} := \{ (X^{\mathcal{I}})_{d_{\mathcal{I}}} \sqsubseteq (X^{\mathcal{I}})_{d_{\mathcal{I}}+1} \mid X \subseteq \Delta^{\mathcal{I}}, X \neq \varnothing \},$$

where

$$d_{\mathcal{I}} := \max_{Y\subseteq\Delta^{\mathcal{I}}} d_{Y^{\mathcal{I}}}.$$

Note that $\mathcal{X}_\mathcal{I}$ is a set of valid GCIs of $\mathcal{I}$, because for each $X \subseteq \Delta^\mathcal{I}$, $X \neq \varnothing$ it is true that

$$((X^\mathcal{I})_{d_\mathcal{I}})^\mathcal{I} = X^{\mathcal{I}\mathcal{I}} = ((X^\mathcal{I})_{d_\mathcal{I}+1})^\mathcal{I}.$$

As already sketched in Section 4.3 the following claims hold.

5.2.29 Lemma *Let $\mathcal{I}$ be a finite interpretation. Then for each $Y \subseteq \Delta^\mathcal{I}$*

i. $\mathcal{X}_\mathcal{I} \models ((Y^\mathcal{I})_k \sqsubseteq (Y^\mathcal{I})_{k+1})$ *is true for all $k \geqslant d_\mathcal{I}$, and*

ii. $\mathcal{X}_\mathcal{I} \models ((Y^\mathcal{I})_k \sqsubseteq Y^\mathcal{I})$ *for all $k \geqslant d_\mathcal{I}$.*

Proof For the first claim we observe that if $Y = \varnothing$, then $Y^\mathcal{I} = \bot$ and nothing remains to be shown. Therefore, let $Y \neq \varnothing$. We shall show the claim by induction over k.

For $k = d_\mathcal{I}$ the claim is trivial, as $((Y^\mathcal{I})_{d_\mathcal{I}} \sqsubseteq (Y^\mathcal{I})_{d_\mathcal{I}+1}) \in \mathcal{X}_\mathcal{I}$. For the step-case $k > d_\mathcal{I}$ assume that

$$\mathcal{X}_\mathcal{I} \models ((Z^\mathcal{I})_{k-1} \sqsubseteq (Z^\mathcal{I})_k) \tag{5.10}$$

is true for all $Z \subseteq \Delta^\mathcal{I}$. Since $Y^\mathcal{I}$ is expressible in terms of $M_\mathcal{I}$, and $Y \neq \varnothing$, there exist $U \subseteq N_C$ and $\Pi \subseteq N_R \times \mathfrak{P}(\Delta^\mathcal{I})$ such that

$$Y^\mathcal{I} \equiv \bigsqcap U \sqcap \bigsqcap_{(r,Z)\in\Pi} \exists r.Z^\mathcal{I}.$$

From Lemma 4.3.10 we obtain

$$(Y^\mathcal{I})_k \equiv \bigsqcap U \sqcap \bigsqcap_{(r,Z)\in\Pi} \exists r.(Z^\mathcal{I})_{k-1}.$$

By the induction hypothesis (5.10) we obtain

$$\begin{aligned}(Y^\mathcal{I})_k &\sqsubseteq \bigsqcap U \sqcap \bigsqcap_{(r,Z)\in\Pi} \exists r.(Z^\mathcal{I})_k \\ &\equiv (\bigsqcap U \sqcap \bigsqcap_{(r,Z)\in\Pi} \exists r.Z^\mathcal{I})_{k+1} \\ &\equiv (Y^\mathcal{I})_{k+1}\end{aligned}$$

again using Lemma 4.3.10. This completes the induction step and the first claim is shown.

Let $k \geqslant d_\mathcal{I}$. We now show the second claim, namely

$$\mathcal{X}_\mathcal{I} \models ((Y^\mathcal{I})_k \sqsubseteq Y^\mathcal{I})$$

for $Y \subseteq \Delta^{\mathcal{I}}$. The case $Y = \varnothing$ is again trivial, so let $Y \neq \varnothing$. Let $\mathcal{J}$ be a finite interpretation such that $\mathcal{J} \models \mathcal{X}_{\mathcal{I}}$. Then the first claim yields

$$((Y^{\mathcal{I}})_k)^{\mathcal{J}} \subseteq ((Y^{\mathcal{I}})_{k+1})^{\mathcal{J}} \subseteq ((Y^{\mathcal{I}})_{k+2})^{\mathcal{J}} \subseteq \dots \tag{5.11}$$

From Lemma 4.3.9 we obtain the existence of some $\ell \in \mathbb{N}_0$ such that $((Y^{\mathcal{I}})_\ell)^{\mathcal{J}} = (Y^{\mathcal{I}})^{\mathcal{J}}$ is true. In particular, $((Y^{\mathcal{I}})_{k+\ell})^{\mathcal{J}} = (Y^{\mathcal{I}})^{\mathcal{J}}$, therefore $((Y^{\mathcal{I}})_k)^{\mathcal{J}} \subseteq (Y^{\mathcal{I}})^{\mathcal{J}}$, and thus $\mathcal{J} \models ((Y^{\mathcal{I}})_k \sqsubseteq Y^{\mathcal{I}})$. Since $\mathcal{EL}^\bot$ has the finite-model property, we obtain $\mathcal{X}_{\mathcal{I}} \models ((Y^{\mathcal{I}})_k \sqsubseteq Y^{\mathcal{I}})$ as required. □

Note that Equation (5.11) can be sated more precisely as

$$((Y^{\mathcal{I}})_k)^{\mathcal{J}} = ((Y^{\mathcal{I}})_{k+1})^{\mathcal{J}} = ((Y^{\mathcal{I}})_{k+2})^{\mathcal{J}} = \dots$$

because $(Y^{\mathcal{I}})_{k+i+1} \sqsubseteq (Y^{\mathcal{I}})_{k+i}$ for all $i \in \mathbb{N}_{>0}$.

Now let $\mathcal{D}$ be a confident base of $\mathrm{Th}_c(\mathcal{I})$. We then can partition $\mathcal{D} = \mathcal{B} \cup \mathcal{C}$ such that $\mathcal{B} \subseteq \mathrm{Th}(\mathcal{I})$ and $\mathcal{C} \cap \mathrm{Th}(\mathcal{I}) = \varnothing$, i. e. $\mathcal{B}$ contains all valid GCIs of $\mathcal{D}$, and $\mathcal{C}$ contains everything else. Without loss of generality we can assume that $\mathcal{B}$ contains only GCIs of the form $E \sqsubseteq E^{\mathcal{II}}$. To construct an $\mathcal{EL}^\bot$ base out of $\mathcal{D}$ we can now proceed as described in the following theorem.

5.2.30 Theorem *Let $\mathcal{I}$ be a finite interpretation, let $c \in [0,1]$ and let $\mathcal{D} = \mathcal{B} \cup \mathcal{C}$ be a finite confident base of $\mathrm{Th}_c(\mathcal{I})$, such that $\mathcal{B} \subseteq \mathrm{Th}(\mathcal{I})$, $\mathcal{C} \cap \mathrm{Th}(\mathcal{I}) = \varnothing$, and $\mathcal{B}$ contains only GCIs of the form $E \sqsubseteq E^{\mathcal{II}}$. Define*

$$d := \max\{\, d_{\mathcal{I}}, \max\{\, d_E \mid (E \sqsubseteq F) \in \mathcal{D} \,\} \,\},$$

and

$$\mathcal{B}' := \{\, E_d \sqsubseteq (E^{\mathcal{II}})_d \mid (E \sqsubseteq E^{\mathcal{II}}) \in \mathcal{B} \,\} \cup \{\, C_d \sqsubseteq (C^{\mathcal{II}})_d \mid (C \sqsubseteq D) \in \mathcal{C} \,\},$$
$$\mathcal{C}' := \{\, (C^{\mathcal{II}})_d \sqsubseteq (D^{\mathcal{II}})_d \mid (C \sqsubseteq D) \in \mathcal{C} \,\}.$$

Then the following statements hold:

i. $\mathcal{B}' \cup \mathcal{X}_{\mathcal{I}} \subseteq \mathrm{Th}(\mathcal{I})$, $\mathcal{C}' \subseteq \mathrm{Th}_c(\mathcal{I})$ *and* $\mathcal{B}' \cup \mathcal{C}' \cup \mathcal{X}_{\mathcal{I}} \models \mathcal{C}$.

ii. $\mathcal{B}' \cup \mathcal{C}' \cup \mathcal{X}_{\mathcal{I}} \models \mathcal{B}$.

In particular, $\mathcal{B}' \cup \mathcal{C}' \cup \mathcal{X}_{\mathcal{I}}$ is a finite confident $\mathcal{EL}^\bot$ base of $\mathrm{Th}_c(\mathcal{I})$.

Proof Clearly, $\mathcal{B}' \cup \mathcal{X}_{\mathcal{I}}$ is a set of valid GCIs. To see that $\mathcal{C}'$ only contains GCIs with high confidence, let $(C \sqsubseteq D) \in \mathcal{C}$ with $|C^{\mathcal{I}}| \neq \varnothing$. Then

$$\mathrm{conf}_{\mathcal{I}}(C \sqsubseteq D) = \mathrm{conf}_{\mathcal{I}}(C^{\mathcal{II}} \sqsubseteq D^{\mathcal{II}})$$

$$
\begin{aligned}
&= \frac{|(C^{\mathcal{II}} \sqcap D^{\mathcal{II}})^{\mathcal{I}}|}{|C^{\mathcal{III}}|} \\
&= \frac{|((C^{\mathcal{II}})_d \sqcap (D^{\mathcal{II}})_d)^{\mathcal{I}}|}{|((C^{\mathcal{II}})_d)^{\mathcal{I}}|} \\
&= \mathrm{conf}_{\mathcal{I}}((C^{\mathcal{II}})_d \sqsubseteq (D^{\mathcal{II}})_d).
\end{aligned}
$$

Clearly, if $C^{\mathcal{I}} = \varnothing$, then $((C^{\mathcal{II}})_d)^{\mathcal{I}} = \varnothing$ and thus

$$\mathrm{conf}_{\mathcal{I}}(C \sqsubseteq D) = 1 = \mathrm{conf}_{\mathcal{I}}((C^{\mathcal{II}})_d \sqsubseteq (D^{\mathcal{II}})_d).$$

Since $\mathcal{C} \subseteq \mathrm{Th}_c(\mathcal{I})$, we therefore obtain $\mathcal{C}' \subseteq \mathrm{Th}_c(\mathcal{I})$ as required.

We now show that $\mathcal{B}' \cup \mathcal{C}' \cup \mathcal{X}_{\mathcal{I}} \models \mathcal{C}$. For this let $(C \sqsubseteq D) \in \mathcal{C}$. Then we have

$$
\begin{aligned}
&\varnothing \models (C \sqsubseteq C_d) \\
&\mathcal{B}' \models (C_d \sqsubseteq (C^{\mathcal{II}})_d) \\
&\mathcal{C}' \models ((C^{\mathcal{II}})_d \sqsubseteq (D^{\mathcal{II}})_d) \\
&\mathcal{X}_{\mathcal{I}} \models ((D^{\mathcal{II}})_d \sqsubseteq D^{\mathcal{II}}) \\
&\varnothing \models (D^{\mathcal{II}} \sqsubseteq D)
\end{aligned}
$$

using Lemma 5.2.29 for the second to last statement. Therefore $\mathcal{B}' \cup \mathcal{C}' \cup \mathcal{X}_{\mathcal{I}} \models (C \sqsubseteq D)$ as required.

We consider the second claim, namely that $\mathcal{B}' \cup \mathcal{C}' \cup \mathcal{X}_{\mathcal{I}} \models \mathcal{B}$. To this end, let $(E \sqsubseteq E^{\mathcal{II}}) \in \mathcal{B}$. Then

$$
\begin{aligned}
&\varnothing \models (E \sqsubseteq E_d) \\
&\mathcal{B}' \models (E_d \sqsubseteq (E^{\mathcal{II}})_d) \\
&\mathcal{X}_{\mathcal{I}} \models ((E^{\mathcal{II}})_d \sqsubseteq E^{\mathcal{II}})
\end{aligned}
$$

using Lemma 5.2.29 for the last statement. Therefore,

$$\mathcal{B}' \cup \mathcal{C}' \cup \mathcal{X}_{\mathcal{I}} \models (E \sqsubseteq E^{\mathcal{II}})$$

as required. □

A drawback of this construction is that the set $\mathcal{X}_{\mathcal{I}}$ can be exponentially large in the size of $\Delta^{\mathcal{I}}$, and so can be the $\mathcal{EL}^{\bot}$ base as described above. Therefore, even if the original base $\mathcal{D}$ was small, the described unravelling can transfer it into a much larger base. It is not known to the author whether this blowup is intrinsic to the task of transferring $\mathcal{EL}^{\bot}_{\mathrm{gfp}}$ bases into $\mathcal{EL}^{\bot}$ bases, or whether it can be avoided by a different approach.

5.3. GCIs with High Confidence from DBpedia

We have started this chapter by evaluating the results by Baader and Distel in a practical scenario. During this evaluation we have found that the presence of errors in the data can impair the usefulness of this approach, and from this we have motivated our study of GCIs with high confidence. In this section, we want to apply our findings about finite confident bases of $\mathrm{Th}_c(\mathcal{I})$ to the interpretation $\mathcal{I}_{\mathrm{DBpedia}}$ we used before, and we want to see in how far the problem of errors in $\mathcal{I}_{\mathrm{DBpedia}}$ can be alleviated by using GCIs with high confidence.

Let us assume that an ontology engineer wants to use the approach of considering GCIs with high confidence to extract GCIs from some finite interpretation $\mathcal{I}$. Then what she has to do is to consider GCIs in $\mathrm{Lux}(\mathcal{I}, c)$ for a suitable choice of c and to decide whether these GCIs should be added to the knowledge base or not. To make this a practical approach, the set $\mathrm{Lux}(\mathcal{I}, c)$ should not be too large, and the GCIs contained in there should not be incomprehensible.

We want to examine how this approach performs for our data set $\mathcal{I}_{\mathrm{DBpedia}}$. To this end, we want to conduct several experiments, namely

i. We want to examine the set $\mathrm{Lux}(\mathcal{I}_{\mathrm{DBpedia}}, 0.95)$ in detail, i. e. we want to show how large this set is and which GCIs are contained in it. Moreover, we shall discuss how the ontology engineer proceeds in deciding whether the GCIs contained in this set are true or not.

ii. We want to compare the sizes of the sets $\mathrm{Conf}(\mathcal{I}_{\mathrm{DBpedia}}, c)$ and $\mathrm{Lux}(\mathcal{I}_{\mathrm{DBpedia}}, c)$ to see in how far we can remove redundancies from $\mathrm{Conf}(\mathcal{I}_{\mathrm{DBpedia}}, c)$ by using $\mathrm{Lux}(\mathcal{I}_{\mathrm{DBpedia}}, c)$ instead. To this end, we shall compute for all $c \in \{ 0.0, 0.01, \ldots, 0.99 \}$ the size of the sets $\mathrm{Conf}(\mathcal{I}_{\mathrm{DBpedia}}, c)$ and $\mathrm{Lux}(\mathcal{I}_{\mathrm{DBpedia}}, c)$ to see how the cardinalities of these sets depends on the choice of the parameter c.

iii. Finally, we want to consider the size of the canonical base of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}_{\mathrm{DBpedia}}})$ for $c \in \{ 0.0, 0.01, \ldots, 0.99 \}$ to see how large this set can be. Recall that $\prod \mathrm{Can}(\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}_{\mathrm{DBpedia}}}))$ is a finite base of $\mathrm{Th}_c(\mathcal{I})$, and the number of GCIs in this base can be considered as a "small" upper limit of how many GCIs are needed to represent $\mathrm{Th}_c(\mathcal{I})$. Intuitively, if we decrease the value for c, we would expect that the number of GCIs contained in the canonical base decreases as well. We shall see how far this is true for $\mathcal{I}_{\mathrm{DBpedia}}$.

The experimental results presented in this section have been published previously in [32].

5.3.1. Computing Confident Bases of $\mathrm{Th}_c(\mathcal{I}_{\mathbf{DBpedia}})$ for $c = 0.95$

Recall that we had constructed $\mathcal{I}_{\mathrm{DBpedia}}$ from the DBpedia data set by extracting all individuals that are in a child-relationship in DBpedia, either as a parent or as a child. Recall that since Wikipedia (from which DBpedia extracts its data) only contains articles about

"famous" persons, we can consider $\mathcal{I}_{\text{DBpedia}}$ as an interpretation that contains all properties about the child-relations between *famous* persons. In particular, if an element of $\mathcal{I}_{\text{DBpedia}}$ does not have a child-successor in $\mathcal{I}_{\text{DBpedia}}$ (and this is not an error), then this does not necessarily mean that the corresponding person does or did not have children – it only means that the children were not famous enough to deserve their own Wikipedia articles.

In the following we want to examine which GCIs we have to consider in addition to the base $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ of $\mathcal{I}_{\text{DBpedia}}$ we had computed in Section 5.1, if we want to consider GCIs which have a confidence in $\mathcal{I}_{\text{DBpedia}}$ of at least 0.95. Thus, let $c = 0.95$. We then can compute $\text{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ to be

$$\begin{aligned}\{\, &\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person},\\ &\mathsf{Place} \sqsubseteq \mathsf{PopulatedPlace},\\ &\exists \mathsf{child}.\exists \mathsf{child}.\top \sqcap \exists \mathsf{child}.\mathsf{OfficeHolder}\\ &\quad \sqsubseteq \exists \mathsf{child}.(\mathsf{OfficeHolder} \sqcap \exists \mathsf{child}.\top)\,\}\end{aligned}$$

Note that the actual GCIs computed are much more complex, since all concept descriptions contained in $\text{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ actually have to be model-based most-specific concept descriptions. For readability, we have removed parts of the concept descriptions which are already entailed by the base $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$, so that the GCIs thus obtained are still equivalent to the original ones. Also notice that $\text{Lux}(\mathcal{I}_{\text{DBpedia}}, c) = \text{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$.

The first observation is that $\text{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$ is small compared to the size of $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$, which has 1252 elements. Thus, our potential ontology engineer only has to consider three more GCIs. Moreover, as we shall see later, accepting some of the above GCIs may even lead to other GCIs in $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ to become dispensable, and thus she does not have to consider those separately.

Let us now consider these three GCIs in detail. The first one we had already seen in Section 5.1, and there we had argued that this is actually true, since the 4 counterexamples contained in $\mathcal{I}_{\text{DBpedia}}$ were only due to errors.

The GCI $\mathsf{Place} \sqsubseteq \mathsf{PopulatedPlace}$ also sounds convincing: note that places appear in $\mathcal{I}_{\text{DBpedia}}$ only due to the fact that they are collected from Wikipedia Infoboxes of articles which contain an entry for children, i. e. which are persons. Since these places occur in the child-entries of infoboxes, they likely name the places of birth of the corresponding children, and those places are usually populated. Indeed, the only counterexamples for $\mathsf{Place} \sqsubseteq \mathsf{PopulatedPlace}$ is Greenwich_Village, representing the corresponding district of Manhattan, New York, which is certainly populated. Thus also this counterexample is erroneous and we accept the GCI as being true (in the domain represented by $\mathcal{I}_{\text{DBpedia}}$).

On the other hand, the last GCI

$$\exists \mathsf{child}.\exists \mathsf{child}.\top \sqcap \exists \mathsf{child}.\mathsf{OfficeHolder} \sqsubseteq \exists \mathsf{child}.(\mathsf{OfficeHolder} \sqcap \exists \mathsf{child}.\top)$$

looks too specific. The only counterexample contained in $\mathcal{I}_{\text{DBpedia}}$ is the element

Pierre_Samuel_du_Pont_de_Nemours

representing the french government official Pierre Samuel du Pont de Nemours. He had two sons, namely Victor Marie du Pont and Eleuthère Irénée du Pont. The former became a french diplomat and is therefore listed as OfficeHolder in $\mathcal{I}_{\text{DBpedia}}$. Although he had four children, none of them were famous enough to receive their own Wikipedia articles. On the other hand, Eleuthère Irénée du Pont became a famous american industrial (founder of the *DuPont* company) and had several famous children which are listed in $\mathcal{I}_{\text{DBpedia}}$. Thus, given our understanding of the child-relation in $\mathcal{I}_{\text{DBpedia}}$ we can accept this counterexample as being valid, and we therefore reject this GCI.

We have accepted the GCIs

$$\mathcal{B} := \{\, \exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}, \mathsf{Place} \sqsubseteq \mathsf{PopulatedPlace} \,\}$$

as being valid although $\mathcal{I}_{\text{DBpedia}}$ contains counterexamples for them. It would now be interesting to know what happened if we would include those two GCIs in a computation of a base of $\mathcal{I}_{\text{DBpedia}}$. For this, note that

$$\mathrm{pr}_{M_{\mathcal{I}_{\text{DBpedia}}}}(\mathcal{B}) = \{\, \{\, \exists \mathsf{child}.\top \,\} \to \{\, \mathsf{Person} \,\}, \{\, \mathsf{Place} \,\} \to \{\, \mathsf{PopulatedPlace} \,\} \,\}.$$

If we now compute

$$\mathcal{L} := \mathrm{Can}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}}, \mathrm{pr}_{M_{\mathcal{I}_{\text{DBpedia}}}}(\mathcal{B}) \cup \mathcal{S}_{\mathcal{I}_{\text{DBpedia}}})$$

then we obtain by Theorem 5.2.26 that $\bigcap \mathcal{L} \cup \mathcal{B}$ is complete for

$$\mathrm{Th}(\mathcal{I}_{\text{DBpedia}}) \cup \{\, \exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}, \mathsf{Place} \sqsubseteq \mathsf{PopulatedPlace} \,\},$$

and is thus a base of it. The set $\mathcal{L}$ now contains 1245 GCIs, and thus $\bigcap \mathcal{L} \cup \mathcal{B}$ contains 1247 GCIs. Comparing this to the 1252 GCIs we can see that some of the GCIs in $\mathcal{B}_{\mathcal{I}_{\text{DBpedia}}}$ indeed became dispensable, although it were not that many.

5.3.2. Sizes of Finite Bases of $\mathrm{Th}_c(\mathcal{I}_{\text{DBpedia}})$

We have seen that $\mathrm{Conf}(\mathcal{I}_{\text{DBpedia}}, 0.95)$ only contains three GCIs. This suggests that the overhead caused by the approach of considering GCIs with high confidence is rather negligible. Moreover, we have also seen that we can reduce the size of bases of $\mathcal{I}_{\text{DBpedia}}$ if we include GCIs from $\mathrm{Conf}(\mathcal{I}_{\text{DBpedia}}, 0.95)$ as background knowledge, thus effectively reducing the number of GCIs our ontology engineer has to consider.

In this section we examine these observations on a larger scale. For this we shall investigate the sizes of the sets $\mathrm{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ and $\mathrm{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$ for varying values of c. From this we shall see how the overhead of considering GCIs with high confidence depends on the choice of the parameter c, and how much we can save by considering $\mathrm{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$ over $\mathrm{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$.

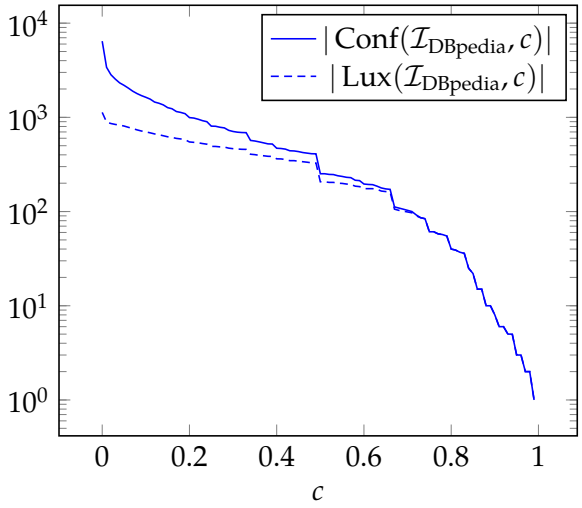


Figure 5.6.: Size of $\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ and $\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$ for all $c \in V$

The Sizes of $\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ and $\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$

We computed $|\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)|$ and $|\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)|$ for all $c \in \{ 0.0, 0.01, 0.02, \ldots, 0.99 \}$. To achieve this, we use

$$|\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)| = |\operatorname{Conf}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}}, c)|,$$
$$|\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)| = |\operatorname{Lux}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}}, c)|.$$

and can thus conduct the computations directly in $\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}}$. The results of these computations are shown in Figure 5.6. Note that the y-axis is scaled logarithmically.

From this picture we can see that for high values of c the amount of GCIs which have to be considered in $\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ and $\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$ is negligible. Even for $c = 0.86$ the set $\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ contains only 15 GCIs. Of course, this observation is per se only valid for $\mathcal{I}_{\text{DBpedia}}$. Since it originates from real-world data one could assume that the same behavior of $|\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)|$ and $|\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)|$ can be expected for other non-artificial datasets.

What we also observe is that for values $c \geqslant 0.73$, $\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ and $\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$ are actually the same sets, i. e. the optimization provided by $\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$ does not take effect for such values of c. Since one usually considers only large values of c the use of $\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$ over $\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ seems questionable.

The Size of $\operatorname{Can}(\operatorname{Th}_c(\mathbb{K}_{\mathcal{I}}))$

Instead of computing confident bases of $\operatorname{Th}_c(\mathcal{I}_{\text{DBpedia}})$ by independently computing bases of $\mathcal{I}_{\text{DBpedia}}$ and conjoining them to $\operatorname{Conf}(\mathcal{I}_{\text{DBpedia}}, c)$ or $\operatorname{Lux}(\mathcal{I}_{\text{DBpedia}}, c)$, we can also utilize

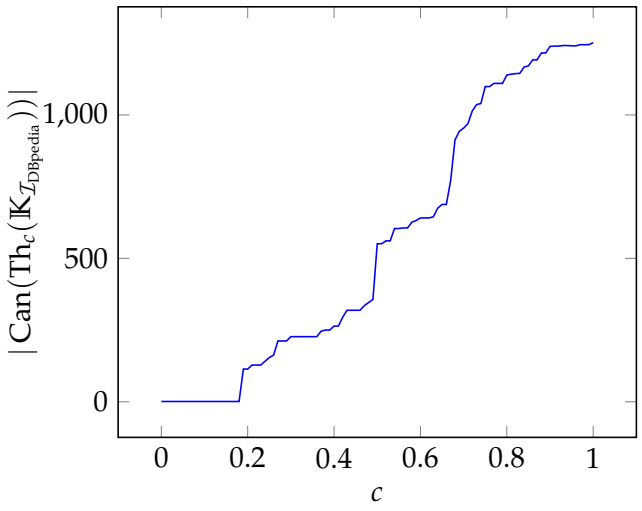


Figure 5.7.: Size of $\mathrm{Can}(\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}_{\mathrm{DBpedia}}}))$ for all $c \in \{0, 0.01, \ldots, 0.99\}$

Theorem 5.2.22. In this theorem, we compute confident bases $\mathcal{L}$ of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}_{\mathrm{DBpedia}}})$, and then $\bigcap \mathcal{L}$ is a finite confident base of $\mathrm{Th}_c(\mathcal{I}_{\mathrm{DBpedia}})$. An advantage of this approach is that bases of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}_{\mathrm{DBpedia}}})$ can be smaller then bases $\mathcal{B}$ of $\mathrm{Th}(\mathcal{I}_{\mathrm{DBpedia}})$ together with $\mathrm{Conf}(\mathcal{I}_{\mathrm{DBpedia}}, c)$, because in the latter case GCIs from $\mathcal{B}$ could already be entailed by GCIs from $\mathrm{Conf}(\mathcal{I}_{\mathrm{DBpedia}}, c)$. Some of these redundancies could already exist on the level of implications, and those could be removed if one computes bases of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$.

In particular, if we compute the canonical base of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$, then all those redundancies are removed. Of course, other dependencies may remain, but the size of the canonical base of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$ may serve as an upper bound on the number of GCIs one needs to represent $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$. On the other hand, considering the canonical base of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}})$ may result in bases which are not confident anymore, i. e. they contain GCIs whose confidence is not above c.

Let us now see how the size of the canonical base changes for all $c \in \{0, 0.01, 0.02, \ldots, 0.99\}$. The results are shown in Figure 5.7.

A first observation is that with decreasing values of c, the size of $\mathrm{Can}(\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}_{\mathrm{DBpedia}}}))$ seems to decrease as well. This is indeed true, except for the case $|\mathrm{Can}(\mathrm{Th}_{0.95}(\mathbb{K}_{\mathcal{I}_{\mathrm{DBpedia}}}))| = 1241$ and $|\mathrm{Can}(\mathrm{Th}_{0.94}(\mathbb{K}_{\mathcal{I}_{\mathrm{DBpedia}}}))| = 1242$. Thus we can observe that with decreasing c, the sets $\mathrm{Th}_c(\mathbb{K}_{\mathcal{I}_{\mathrm{DBpedia}}})$ get "simpler" in the sense that fewer implications are necessary to represent them. If the same is true for bases of $\mathrm{Th}_c(\mathcal{I}_{\mathrm{DBpedia}})$ is not clear, though, as it is not quite clear how to compute finite bases of sets of GCIs.

Clearly, the farther away c is from 1, the more the set $\mathrm{Th}_c(\mathcal{I}_{\mathrm{DBpedia}})$ departs from $\mathrm{Th}(\mathcal{I}_{\mathrm{DBpedia}})$. There are three values for c where this becomes especially apparent: for $c \in \{0.18, 0.54, 0.66\}$ the curve depicted in Figure 5.7 shows a rather steep decline. Indeed,

these values of c are not arbitrary, since we can associate some special observations with them:

i. For $c \leqslant 0.18$ the implication $\varnothing \to M_{\mathcal{I}_{\text{DBpedia}}}$ is entailed by $\text{Th}_c(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$, resulting in a singleton canonical base.

ii. For $0.18 < c \leqslant 0.54$ the implication $\{ \exists \text{child}.\top \} \to \{ \exists \text{child}.\text{Person} \}$ is contained in $\text{Th}_c(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$, eliminating a large number of special cases.

iii. For $0.54 < c \leqslant 0.66$ the implication $\varnothing \to \{ \text{Person} \}$ is contained in $\text{Th}_c(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$, also making a number of other GCIs dispensable.

One can observe that the implications found in the last two cases also account for a rather drastic decline of the size of the canonical base on their own: the canonical base of $\text{Th}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$ contains 1252 elements, but the canonical base of

$$\text{Th}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}}) \cup \{ \varnothing \to \{ \text{Person} \} \}$$

contains only 1210 implications. If we also add the implication $\{ \exists \text{child}.\top \} \to \{ \exists \text{child}.\text{Person} \}$, then the size of the canonical base drops to 1163.

Notice that this is actually not very surprising: the implications

$$\{ \exists \text{child}.\top \to \{ \exists \text{child}.\text{Person} \} \}, \quad \varnothing \to \{ \text{Person} \}$$

can be considered rather *general*, because they are applicable to a lot of elements in $\mathcal{I}_{\text{DBpedia}}$. Because of this generality they can make a lot of other implications redundant, causing the size of the canonical base to be reduced noticeably.

Indeed, we could argue that whenever the size of the canonical base of $\text{Th}_c(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$ is reduced drastically, then it is very likely that a general GCI has been found, in the sense that the confidence threshold is now so low that it is accepted. To this end recall that we had already noticed that the size of the canonical base of $\text{Th}_c(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$ can be seen as some kind of indication how *complex* this set is: the less implications are contained in the canonical base, the less implications are necessary to represent $\text{Th}_c(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$ and thus the *simpler* this set is, from a logical point of view. Therefore, if we can observe for $c_1 > c_2$ a steep decline in the size of the canonical base of $\text{Th}_{c_2}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$ compared to the size of the canonical base of $\text{Th}_{c_1}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$, it is legitimate to assume that then $\text{Th}_{c_2}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$ contains new implications which make a lot of implications in the canonical base of $\text{Th}_{c_1}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$ dispensable. Those implications then can considered to be very general, since they need to account for the entailment, which in $\text{Th}_{c_1}(\mathbb{K}_{\mathcal{I}_{\text{DBpedia}}})$ has only been achieved by a larger number of implications.

Therefore, the way the size of $\text{Can}(\text{Th}_c(\mathbb{K}_{\mathcal{I}}))$ depends on the choice of c can indicate which kinds of GCIs have been accepted with the current confidence threshold c. This can especially be interesting when looking for general patterns in the interpretation $\mathcal{I}$, or

when it is not yet clear which value for the confidence threshold to choose. On the other hand, it is also quite obvious that the preceding argumentation is not formal, and that this approach should be seen as a heuristics.

CHAPTER 6

Exploration by Confidence

Recall that we have introduced GCIs with high confidence as an approach to extract terminological knowledge from erroneous data. For this approach we have seen in Section 5.3.1 that the thus obtained GCIs require an additional validation step, i. e. an expert has to verify whether the extracted GCIs, whose confidence in the data $\mathcal{I}$ is above a pre-chosen threshold $c \in [0, 1]$, are indeed valid in the domain of interest.

This additional validation step can be very costly, and thus it should be avoided as much as possible. On the other hand, we have also seen in Section 5.3.1 that as soon as some GCI has been confirmed, others may be entailed by it, and a manual validation is no longer necessary. This observation may indeed save a lot of work.

Even more, utilizing this observation may save computation time as well: if during the computation of bases of $\mathrm{Th}_c(\mathcal{I})$ an expert already rejects a GCI and confirms that some counterexamples contained in $\mathcal{I}$ are indeed correct, then all GCIs which are falsified by these counterexamples can also be rejected, irrespective of whether their confidence in $\mathcal{I}$ is above c, or not. It is thus desirable to allow expert interaction already during the time of the computation of bases of $\mathrm{Th}_c(\mathcal{I})$, and not only at the end of the computation, as a separate validation step.

There is also another issue that requires expert interaction, and which has already been addressed by Distel [41]: the data $\mathcal{I}$ may not only contain errors, but it may also be *incomplete* in the sense that it lacks certain counterexamples, meaning that some GCIs are valid in $\mathcal{I}$, even so they are not valid in the domain of interest. If $\mathcal{I}$ contains errors, it may additionally happen that some invalid GCIs $C \sqsubseteq D$ are only falsified in $\mathcal{I}$ by erroneous counterexamples, and that correct counterexamples are not present in $\mathcal{I}$. In such a case, the GCI $C \sqsubseteq D$ would be accepted (since all counterexamples are erroneous) although it is not valid. Here an expert could, as soon as $C \sqsubseteq D$ would be computed as an element of a base, provide correct counterexamples, inhibiting this GCI from being included in a base. Such counterexamples would also affect the following computation, as all GCIs invalidated by it would also be rejected.

The approach followed by Distel to solve this problem is to adapt attribute exploration

from its original setting of implications and formal contexts to the setting of GCIs and finite interpretations. Recall that the attribute exploration algorithm, which we had discussed previously in Section 2.5, solves a problem which is very similar to the one sketched above: given a finite formal context $\mathbb{K}$, we want to compute a base of the domain which is represented by $\mathbb{K}$. However, it may be the case that $\mathbb{K}$ does not completely represent the domain we are interested in, i. e. some implications which are not valid in our domain actually hold in $\mathbb{K}$. Then attribute exploration provides an interactive base computation procedure which, as soon as an implication is computed, presents it to the expert and asks for validation. If the expert confirms this implication, then the implication is added to the base. If the expert does not confirm this implication, she has to provide a counterexample, which is then included in $\mathbb{K}$. At the end, the attribute exploration algorithm yields a base of the domain we are interested in, and not only one for the formal context $\mathbb{K}$.

Distel's generalizations of attribute exploration to the setting of GCIs and finite interpretation are called *model exploration* and *ABox exploration*. Their abstract behavior is very similar to the one of attribute exploration: during the computation of a base of $\mathcal{I}$, GCIs $C \sqsubseteq D$ are asked to the expert. If confirmed, they are added to the base. If rejected, the expert has to provide a counterexample which is being added to $\mathcal{I}$. Upon termination, the algorithm yields a base of the domain which is represented by the expert.

Model exploration and ABox exploration differ in the way counterexamples are provided. In model exploration, counterexamples are directly added to the interpretation $\mathcal{I}$, whereas in ABox exploration the counterexamples are being added to a separate ABox. The latter variant is much more user-friendly from the point of view of how to specify counterexamples, but the resulting exploration algorithm is much more technical and complicated. Therefore, we shall concentrate in this work on model exploration only.

In the following two chapters we want to generalize model exploration to our setting of extracting GCIs with high confidence from finite interpretations. More precisely, what we want to obtain is an algorithm that computes bases of $\mathrm{Th}_c(\mathcal{I})$, and, as soon as a GCI for the base is computed, asks the expert for validation. As above, the expert may confirm or reject this GCI, where in the latter case she has to provide a counterexample. Upon termination, the algorithm should yield a base of the domain that is represented by the expert, to the extent in which this domain is contained in $\mathrm{Th}_c(\mathcal{I})$. We shall make this more precise in Chapter 7.

The purpose of this chapter is to prepare this generalization of model exploration to GCIs with high confidence by first discussing the analogous problem in the world of formal concept analysis. For this, we want to develop an algorithm for *exploration by confidence*, which not only asks implications that are valid in the current working context, but may just have a high confidence in the original data set.

To this end, we shall first consider in Section 6.1.1 classical attribute exploration from a more abstract point of view. In this consideration, we shall see that we can consider attribute exploration of a formal context $\mathbb{K}$ as an algorithm that "explores" the set $\mathrm{Th}(\mathbb{K})$, in the sense that attribute exploration tries to find out which implications in $\mathrm{Th}(\mathbb{K})$ are

confirmed by an expert and which not. To make this more precise, we shall also introduce a formalization of the up to now only informally treated notion of an *expert*.

Based on the observation that attribute exploration is an algorithm to explore $\mathrm{Th}(\mathbb{K})$, we shall introduce in Section 6.1.2 a generalization of attribute exploration that allows to explore arbitrary sets $\mathcal{L}$ of implications. The main idea is that this *exploration of sets of implications* works in the same way as attribute exploration, namely by asking implications from $\mathcal{L}$ to the expert to try to find out which of the implications in $\mathcal{L}$ are accepted by the expert and which not. However, the algorithm we are going to present in Section 6.1.2 will not completely achieve this goal, but instead will provide us with an *approximative exploration* of $\mathcal{L}$, a notion which we shall make clear then. Intuitively, this means that also implications which are not elements of $\mathcal{L}$, but of $\mathrm{Cn}(\mathcal{L})$, can be asked to the expert, and thus may appear in the output. Still, we shall see in Section 6.2 that this approximative exploration is sufficient to allow us to obtain an algorithm for exploration by confidence. The main idea for this is to use the set $\mathcal{L} = \mathrm{Th}_c(\mathbb{K})$ as the set we want to explore. In this case we shall also see how the approximative exploration provided by the general algorithm can be turned into a proper exploration, i. e. we can ensure in the special case of $\mathcal{L} = \mathrm{Th}_c(\mathbb{K})$ that all implications asked to the expert indeed have high confidence in the original data set.

The results presented in this section have partially been published previously in [30].

6.1. Exploring Sets of Implications

Let M be a finite set and $\mathcal{L} \subseteq \mathrm{Imp}(M)$. In this section we want to discuss how we can turn attribute exploration into an algorithm for *exploring* $\mathcal{L}$. To this end, we shall first give another perspective on the attribute exploration algorithm that we have already met in Section 2.5. Based upon this, we shall introduce in Section 6.1.2 a *generalized attribute exploration* that allows us to explore $\mathcal{L}$. This generalization is however only approximative, in a sense that we shall make clear then.

6.1.1. An Abstract View on Attribute Exploration

Recall that in attribute exploration as introduced in Section 2.5 we assumed that the domain of interest can be represented by a formal context $\mathbb{K}_{\mathsf{back}}$, which we call the *background context* of the exploration. The goal of attribute exploration is then to find a base of $\mathbb{K}_{\mathsf{back}}$, without having direct access to $\mathbb{K}_{\mathsf{back}}$. For this we start with a subcontext $\mathbb{K} = (G, M, I)$ of $\mathbb{K}_{\mathsf{back}}$, the *working context* of the exploration, and a set $\mathcal{K} \subseteq \mathrm{Imp}(M)$ of implications which are valid in $\mathbb{K}_{\mathsf{back}}$, called the set of *known implications*. Then we successively compute implications of the form

$$P \to P''$$

where P is a $\mathcal{K}$-pseudo intent of $\mathbb{K}$, and in particular is closed under the set of currently known implications, but is not an intent of the current working context. Those implications are presented to the expert, who either confirms or rejects them.

We can view this procedure from a more general perspective:[1] given the formal context $\mathbb{K}$, we know that all implications in $\mathrm{Imp}(M)\setminus\mathrm{Th}(\mathbb{K})$ *are not* valid in $\mathbb{K}_{\mathsf{back}}$, and thus are not valid in our domain. On the other hand, all implications in $\mathrm{Th}(\mathbb{K})$ *could* be valid in $\mathbb{K}_{\mathsf{back}}$, depending on whether counterexamples contained in $\mathbb{K}_{\mathsf{back}}$ invalidate implications being valid in $\mathbb{K}$ or not. However, what we are certain of is that all implications in $\mathcal{K}$ *are* valid in our domain, and thus all implications in $\mathrm{Cn}(\mathcal{K})$ are.

Hence, we have the situation that we have three sets of implications, namely the set of *all* implications $\mathrm{Imp}(M)$, the set $\mathrm{Th}(\mathbb{K})$ of *possibly valid* implications, and the set of *certainly valid* implications $\mathrm{Cn}(\mathcal{K})$. These sets are related by

$$\mathrm{Imp}(M) \supseteq \mathrm{Th}(\mathbb{K}) \supseteq \mathrm{Cn}(\mathcal{K}),$$

since $\mathcal{K}$ is supposed to be sound for $\mathbb{K}$. Now the set

$$\mathrm{Th}(\mathbb{K})\setminus\mathrm{Cn}(\mathcal{K})$$

can be seen as the set of *undecided* implications, i. e. the set of those implications which are possibly valid in $\mathbb{K}_{\mathsf{back}}$, but from which we do not know yet whether they indeed are valid in $\mathbb{K}_{\mathsf{back}}$ or not (since we do not have direct access to $\mathbb{K}_{\mathsf{back}}$). Then attribute exploration can be viewed as a *systematic search* through the set $\mathrm{Th}(\mathbb{K})\setminus\mathrm{Cn}(\mathcal{K})$, in the sense that the crucial feature for attribute exploration to work is to be able to compute implications

$$(P \to P'') \in \mathrm{Th}(\mathbb{K})\setminus\mathrm{Cn}(\mathcal{K}), \tag{6.1}$$

provided that $\mathrm{Th}(\mathbb{K})\setminus\mathrm{Cn}(\mathcal{K}) \neq \emptyset$. We shall argue now why this is indeed enough.

Let $(P \to P'') \in \mathrm{Th}(\mathbb{K})\setminus\mathrm{Cn}(\mathcal{K})$. Then the implication $P \to P''$ is proposed to the expert. If she accepts $P \to P''$, then we add this implication to $\mathcal{K}$, and we obtain the following situation:

$$\mathrm{Imp}(M) \supseteq \mathrm{Th}(\mathbb{K}) \supseteq \mathrm{Cn}(\mathcal{K} \cup \{ P \to P'' \}) \supsetneq \mathrm{Cn}(\mathcal{K}).$$

If the expert rejects $P \to P''$, she has to provide a counterexample, i. e. a set $C \subseteq M$ such that $P \subseteq C$ and $P'' \nsubseteq C$. Denote with $\mathbb{K} + C$ the formal context which arises from $\mathbb{K}$ by adding a new object g_C to $\mathbb{K}$ which has exactly the attributes which are contained in C, i. e.

$$(g_C)' := C$$

where the derivation is done in $\mathbb{K} + C$. Then the situation from above evolves into

$$\mathrm{Imp}(M) \supseteq \mathrm{Th}(\mathbb{K}) \supsetneq \mathrm{Th}(\mathbb{K} + C) \supseteq \mathrm{Cn}(\mathcal{K}).$$

[1] This idea is similar to the one of considering the process of attribute exploration as a decreasing sequence of intervals in the lattice of closure systems over M, see [47, pp. 143–145] for more details.

If $\mathrm{Th}(\mathbb{K}) \setminus \mathrm{Cn}(\mathcal{K}) = \varnothing$, then the algorithm terminates. (Note that this indeed has to happen since the base set M is finite.) When this happens, the situation can be described as follows[2]:

$$\begin{aligned}\mathrm{Imp}(M) &\supseteq \mathrm{Th}(\mathbb{K}) \supsetneq \mathrm{Th}(\mathbb{K} + C_1) \supsetneq \cdots \supsetneq \mathrm{Th}(\mathbb{K} + C_1 + \cdots + C_m) \\ &= \mathrm{Cn}(\mathcal{K} \cup \{ P_1 \to P_1'', \ldots, P_n \to P_n'' \}) \supsetneq \cdots \supsetneq \mathrm{Cn}(\mathcal{K})\end{aligned}$$

where $m, n \in \mathbb{N}_{\geqslant 0}$, $C_1, \ldots, C_m, P_1, \ldots, P_n \subseteq M$, and where all derivations are done in $\mathbb{K} + C_1 + \cdots + C_m$. It is then true that

$$\mathrm{Th}(\mathbb{K} + C_1 + \cdots + C_m) = \mathrm{Th}(\mathbb{K}_{\mathsf{back}}) = \mathrm{Cn}(\mathcal{K} \cup \{ P_1 \to P_1'', \ldots, P_n \to P_n'' \})$$

just because

$$\mathrm{Th}(\mathbb{K} + C_1 + \cdots + C_i) \supseteq \mathrm{Th}(\mathbb{K}_{\mathsf{back}}) \supseteq \mathrm{Cn}(\mathcal{K} \cup \{ P_1 \to P_1'', \ldots, P_j \to P_j'' \})$$

for all $1 \leqslant i \leqslant m$ and $1 \leqslant j \leqslant n$. In particular, the set

$$\{ P_1 \to P_1'', \ldots, P_n \to P_n'' \}$$

is a base of $\mathbb{K}_{\mathsf{back}}$ with background knowledge $\mathcal{K}$, as required.

Note that we did not require that the set P from Equation (6.1) is a $\mathcal{K}$-pseudo intent of $\mathbb{K}$. The only necessary requirement is that $P \neq P''$ and that $P \to P''$ does not follow from $\mathcal{K}$. Both these properties are guaranteed by P being a $\mathcal{K}$-pseudo intent of $\mathbb{K}$, but this is not necessary.

On the other hand, if we want to compute the canonical base of $\mathbb{K}_{\mathsf{back}}$ with background knowledge $\mathcal{K}$, then P obviously needs to be a $\mathcal{K}$-pseudo intent of $\mathbb{K}$. However, since in our perception attribute exploration should compute *some* base of $\mathbb{K}_{\mathsf{back}}$, we can see the computation of $\mathrm{Can}(\mathbb{K}_{\mathsf{back}}, \mathcal{K})$ as an optimization, and not as a crucial requirement.

6.1.2. A Generalized Attribute Exploration

The foregone considerations suggest that we can view attribute exploration as an algorithm that allows us to "explore" the set $\mathrm{Th}(\mathbb{K}) \setminus \mathrm{Cn}(\mathcal{K})$ of undecided implications. In this section we want to generalize this view to the setting where the set of undecided implications has a more general form.

To make this more precise, let M be a finite set, $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and let $\mathcal{K} \subseteq \mathrm{Cn}(\mathcal{L})$. As we have viewed the set $\mathrm{Th}(\mathbb{K})$ as the set of possibly valid implications, we can now view the set $\mathcal{L}$ as the set of possibly valid implications. Then the set of undecided implications takes the form

$$\mathcal{L} \setminus \mathrm{Cn}(\mathcal{K})$$

[2] We read $\mathbb{K} + C_1 + \cdots + C_m$ as $(\ldots(\mathbb{K} + C_1) + \ldots) + C_m$ and nothing else.

and we want to obtain an algorithm that guides an expert in finding all valid implications in $\mathcal{L}\setminus \mathrm{Cn}(\mathcal{K})$. More precisely, we want to compute a base of all implications valid in $\mathcal{L}$, using $\mathcal{K}$ as background knowledge. As before, we assume that we do not have direct access to the background context $\mathbb{K}_{\mathsf{back}}$, the formal context that is represented by the expert.

At first sight, one may be tempted to say that exploring $\mathcal{L}$ can be achieved (at least in theory) by considering the formal context

$$\mathbb{K}_{\mathcal{L}} = (\{\, \mathcal{L}(X) \mid X \subseteq M \,\}, M, \ni)$$

which already appeared in Proposition 2.3.7. This proposition tells us that $\mathrm{Th}(\mathbb{K}_{\mathcal{L}}) = \mathrm{Cn}(\mathcal{L})$, and thus one could propose that for exploring $\mathcal{L}$ it would just be sufficient to explore $\mathbb{K}_{\mathcal{L}}$. However, besides the fact that computing $\mathbb{K}_{\mathcal{L}}$ (or a subcontext of it which still has $\mathrm{Cn}(\mathcal{L})$ as its theory) is far from practical, this approach also does not solve our initial problem, because in general $\mathcal{L} = \mathrm{Cn}(\mathcal{L})$ does not hold. This is illustrated by the following example.

6.1.1 Example Let $M = \{\mathsf{a},\mathsf{b},\mathsf{c}\}, \mathcal{K} = \varnothing, \mathcal{L} = \{\{\mathsf{a}\} \to \{\mathsf{b}\}\}$ and suppose that our domain can be represented by the formal context

$\mathbb{K}_{\mathsf{back}}$	a	b	c
g	×		

Then clearly $\mathrm{Th}(\mathbb{K}_{\mathsf{back}}) \cap \mathcal{L} = \varnothing$, and thus all bases of this set contain trivial implications (i. e. implications of the form $P \to P$ for $P \subseteq M$). On the other hand,

$$(\{\mathsf{a},\mathsf{c}\} \to \{\mathsf{b}\}) \in \mathrm{Cn}(\mathcal{L}) \cap \mathrm{Th}(\mathbb{K}_{\mathsf{back}}),$$

and therefore bases of $\mathrm{Th}(\mathbb{K}_{\mathsf{back}}) \cap \mathrm{Cn}(\mathcal{L})$ contain non-trivial implications.

Thus, exploring $\mathcal{L}$ and exploring $\mathrm{Cn}(\mathcal{L})$, using the aforementioned expert and $\mathcal{K}$ as background knowledge, are different tasks. ♢

Intuitively, an algorithm for exploring $\mathcal{L}$ should work as attribute exploration does: if the implication $P \to Q$ is accepted by the expert, then it is added to $\mathcal{K}$. If it is rejected by the expert, then a counterexample C has to be provided by the expert. In this case, all implications from $\mathcal{L}$ for which C is a counterexample are removed. The algorithm terminates if $\mathcal{L}\setminus \mathrm{Cn}(\mathcal{K}) = \varnothing$ (in which case also $\mathrm{Cn}(\mathcal{L})\setminus \mathrm{Cn}(\mathcal{K}) = \varnothing$ holds).

In what follows, we want to make this idea more precise. To this end, we shall first start with a formalization of the notion of an *expert* that we heretofore have used only intuitively.

6.1.2 Definition (Domain Expert) Let M be a set. A *domain expert* on M is a function

$$p\colon \mathrm{Imp}(M) \to \{\top\} \cup \mathfrak{P}(M),$$

where $\top \notin \mathfrak{P}(M)$, and which satisfies the following conditions

i. if $(X \to Y) \in \mathrm{Imp}(M)$ such that $p(X \to Y) = C \neq \top$, then $X \subseteq C$ and $Y \nsubseteq C$, (*p gives counterexamples for false implications*)

ii. if $(U \to V), (X \to Y) \in \mathrm{Imp}(M)$ such that $p(U \to V) = \top, p(X \to Y) = C \neq \top$, then C is not a counterexample for $U \to V$, i. e. either $U \nsubseteq C$ or $V \subseteq C$. (*counterexamples from p do not invalidate confirmed implications*)

We say that *p confirms* $(X \to Y) \in \mathrm{Imp}(M)$ if and only if $p(X \to Y) = \top$. Otherwise, we say that *p rejects* $X \to Y$ and *provides* $C = p(X \to Y)$ *as a counterexample*. Finally, the *theory* $\mathrm{Th}(p)$ of p is the set of all implications over M which are confirmed by p. ♢

We have not specified the notion of a *domain* formally, but instead required that every domain is representable by a formal context. With our formalization of an expert we can now show that experts provide an equally good approach to formalize the notion of a domain as formal contexts do.

6.1.3 Proposition *Let $\mathbb{K} = (G, M, I)$ be a formal context. For each $(A \to B) \in \mathrm{Imp}(M) \setminus \mathrm{Th}(\mathbb{K})$ let $g_{A\to B} \in G$ such that $A \subseteq g'_{A\to B}$ and $B \nsubseteq g'_{A\to B}$. Then the mapping $p_{\mathbb{K}} \colon \mathrm{Imp}(M) \to \{\top\} \cup \mathfrak{P}(M)$ defined by*

$$p_{\mathbb{K}}(X \to Y) := \begin{cases} g_{X\to Y} & (X \to Y) \notin \mathrm{Th}(\mathbb{K}) \\ \top & \text{otherwise} \end{cases}$$

is a domain expert on M, and $\mathrm{Th}(\mathbb{K}) = \mathrm{Th}(p_{\mathbb{K}})$.

Proof Clearly, if $(X \to Y) \notin \mathrm{Th}(\mathbb{K})$, then $p_{\mathbb{K}}(X \to Y) = g_{X\to Y}$ is a counterexample for $X \to Y$. If $(X \to Y) \in \mathrm{Th}(\mathbb{K})$, then for all $g \in G$ it is true that either $X \nsubseteq g'$ or $Y \subseteq g'$. Thus, no counterexample provided by $p_{\mathbb{K}}$ is a counterexample for $X \to Y$. Therefore, counterexamples provided by p do not invalidate confirmed implications. The equality $\mathrm{Th}(\mathbb{K}) = \mathrm{Th}(p_{\mathbb{K}})$ is clear from the definition of $p_{\mathbb{K}}$. ▫

Note that the actual definition of $p_{\mathbb{K}}$ depends on the particular choice of the objects $g_{A\to B}$, and thus $\mathbb{K}$ can give rise to more than one domain expert.

If we have given a domain expert, we can easily construct a formal context from it that has the same theory.

6.1.4 Proposition *Let p be domain expert on a set M. Then the formal context $\mathbb{K}_p = (G, M, \ni)$, where*

$$G := \{\, p(X \to Y) \mid (X \to Y) \in \mathrm{Imp}(M) \setminus \mathrm{Th}(p) \,\}$$

satisfies $\mathrm{Th}(p) = \mathrm{Th}(\mathbb{K}_p)$.

Proof If $(A \to B) \in \mathrm{Th}(p)$, then no counterexample provided by p invalidates $A \to B$. Since the rows of $\mathbb{K}_p$ consist of the counterexamples provided by p, $\mathbb{K}_p$ does not contain a counterexample for $A \to B$ and thus $(A \to B) \in \mathrm{Th}(\mathbb{K}_p)$.

If $p(A \to B) \notin \mathrm{Th}(p)$, then $p(A \to B) \subseteq M$ is a counterexample for $A \to B$ and therefore $(A \to B) \notin \mathrm{Th}(\mathbb{K}_p)$. ▫

In particular, for every domain expert p on M and every formal context $\mathbb{K} = (G, M, I)$ it is true that

$$\mathrm{Th}(p) = \mathrm{Th}(p_{\mathbb{K}_p}),$$
$$\mathrm{Th}(\mathbb{K}) = \mathrm{Th}(\mathbb{K}_{p_{\mathbb{K}}}).$$

In other words, the representation of domains as formal contexts and in terms of domain experts is, from a logical point of view, interchangeable.

With the formal notion of an expert we can reformulate our goal of exploring sets of implications.

6.1.5 Definition (Exploring Sets of Implications) Let M be a finite set, p a domain expert on M, $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and $\mathcal{K} \subseteq \mathrm{Th}(p) \cap \mathrm{Cn}(\mathcal{L})$. Then to *explore* $\mathcal{L}$ *with expert* p *and background knowledge* $\mathcal{K}$ means to compute a base of

$$\mathcal{L} \cap \mathrm{Th}(p)$$

with background knowledge $\mathcal{K}$. ♢

As an approach to find an algorithm that allows us to explore $\mathcal{L}$, we want to suitably adapt the classical attribute exploration algorithm. To this end, we shall first consider a slight reformulation of the attribute exploration algorithm as given in Algorithm 3, and then try to adapt it to our setting of exploring $\mathcal{L}$.

Recall that in Algorithm 3 we have used two special functions, namely for computing the lectically first closed set that is not an intent of a given formal context, and for computing the lectically next closed set of a given closure operator that is not an intent of the current working context. However, a closer inspection of the way these functions are used reveals that attribute exploration can be reformulated using only a function that computes for a given set $A \subseteq M$ the lectically smallest set lectically greater *or equal* to A that is not an intent of the current working context but is closed under a given closure operator. An implementation of this function, called `next-closed-non-closed`, is given in Algorithm 4. This function suffices to implement attribute exploration, and the corresponding listing is also shown in Algorithm 4.

We want to use this reformulation as a starting point to discuss an algorithm for exploring $\mathcal{L}$. To this end, recall how we have considered attribute exploration in Section 6.1.1 as an

Algorithm 4 (Another Implementation of Attribute Exploration)

```
0  define next-closed-non-closed(M, ⩽_M, A, 𝒦 ⊆ Imp(M), ℒ ⊆ Imp(M))
1    ;; computes the lectically smallest element greater or equal to A that is closed
2    ;; under 𝒦 and not closed under ℒ
3
4    if A = nil then
5      return nil
6    else if A = 𝒦(A) and A ≠ ℒ(A) then
7      return A
8    else
9      return next-closed-non-closed(M, ⩽_M, next-closure(M, ⩽_M, A, 𝒦), 𝒦, ℒ)
10   end
11 end
12
13 define explore-attributes(M, ⩽_M, p, 𝕂 = (G, M, I), 𝒦 ⊆ Th(p))
14   i := 0, 𝕂_i := 𝕂, 𝒦_i := 𝒦, P_i := ∅
15
16   forever do
17     P_{i+1} := next-closed-non-closed(M, ⩽_M, P_i, 𝒦_i, (.)''_{𝕂_i})
18     if P_{i+1} = nil exit
19
20     if p(P_{i+1} → (P_{i+1})''_{𝕂_i}) = ⊤ then
21       𝒦_{i+1} := 𝒦_i ∪ { P_{i+1} → (P_{i+1})''_{𝕂_i} }
22       𝕂_{i+1} := 𝕂_i
23     else
24       𝒦_{i+1} := 𝒦_i
25       C := p(P_{i+1} → (P_{i+1})''_{𝕂_i})
26       𝕂_{i+1} := 𝕂_i + C
27     end
28
29     i := i + 1
30   end
31
32   return 𝒦_i
33 end
```

algorithm for exploring $\mathrm{Th}(\mathbb{K})$. Then in iteration i of the attribute exploration algorithm, the set of possibly valid implications is given by $\mathrm{Th}(\mathbb{K}_i)$, where

$$\mathbb{K}_i = \mathbb{K} + C_1 + \cdots + C_{k_i}$$

for some $k_i \in \mathbb{N}, C_1, \ldots, C_{k_i} \subseteq M$. The idea is now that in Algorithm 4 we replace every reference to $\mathrm{Th}(\mathbb{K}_i)$ by a reference to a set $\mathcal{L}_i$ of implications. For this we observe that if a counterexample $C \subseteq M$ is given by the expert in iteration i, then

$$\begin{aligned}\mathrm{Th}(\mathbb{K}_{i+1}) &= \mathrm{Th}(\mathbb{K}_i + C)\\ &= \{\, (A \to B) \in \mathrm{Th}(\mathbb{K}_i) \mid A \not\subseteq C \text{ or } B \subseteq C \,\}.\end{aligned}$$

Correspondingly, we obtain

$$\mathcal{L}_{i+1} = \{\, (A \to B) \in \mathcal{L}_i \mid A \not\subseteq C \text{ or } B \subseteq C \,\}.$$

The result of this simple replacement is shown in Algorithm 5.

However, Algorithm 5 does not exactly yield an algorithm to explore $\mathcal{L}$, using p as domain expert and $\mathcal{K}$ as background knowledge. Indeed, it may be the case that the set $\mathcal{K}_n$ of implications returned by Algorithm 5 contains too many implications, i. e. it may happen that

$$\mathrm{Cn}(\mathcal{K}_n) \setminus \mathrm{Cn}(\mathrm{Th}(p) \cap \mathcal{L}) \neq \varnothing.$$

This is illustrated by the following example[3]

6.1.6 Example Let $M = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$, $a \leqslant_M b \leqslant_M c$, and define an expert p on M by $\mathrm{Th}(p) = \mathrm{Cn}(\{\mathsf{c}\} \to \{\mathsf{a}, \mathsf{b}\})$. Define furthermore $\mathcal{L} = \{\{\mathsf{a}\} \to \{\mathsf{b}\}, \{\mathsf{c}\} \to \{\mathsf{a}\}\}$. Then the implication

$$\{\mathsf{c}\} \to \{\mathsf{a}, \mathsf{b}\},$$

is proposed to the expert p, who confirms it. Thus, this implication is an element of $\mathcal{K}_n$, but is not contained in $\mathrm{Th}(p) \cap \mathcal{L}$. ♢

However, we shall show that Algorithm 5 *approximately explores* $\mathcal{L}$, in the sense that the set $\mathcal{K}_n$ satisfies

$$\mathrm{Cn}(\mathcal{L}) \cap \mathrm{Th}(p) \supseteq \mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathrm{Th}(p) \cap \mathcal{L}). \tag{6.2}$$

In other words, the resulting set of confirmed implications will at least be complete for $\mathrm{Cn}(\mathrm{Th}(p) \cap \mathcal{L})$, and the implications which are not entailed by $\mathrm{Th}(p) \cap \mathcal{L}$ are at least entailed by $\mathrm{Cn}(\mathcal{L}) \cap \mathrm{Th}(p)$. It will turn out that this is indeed sufficient for our considerations of finding an algorithm for exploration by confidence, as we shall see in Section 6.2.2.

[3] This example has been provided by an anonymous reviewer, who also pointed our an error in the original argumentation (as given in [30]).

Algorithm 5 (Exploration of Sets of Implications)

```
0  define explore-implications(M, ⩽_M, p, L ⊆ Imp(M), K ⊆ Th(p))
1    i := 0, L_i := L, K_i := K, P_i := ∅
2
3    forever do
4      P_{i+1} := next-closed-non-closed(M, ⩽_M, P_i, K_i, L_i)
5      if P_{i+1} = nil exit
6
7      if p(P_{i+1} → L_i(P_{i+1})) = ⊤ then
8        K_{i+1} := K_i ∪ { P_{i+1} → L_i(P_{i+1}) }
9        L_{i+1} := L_i
10     else
11       K_{i+1} := K_i
12       C := p(P_{i+1} → L_i(P_{i+1}))
13       L_{i+1} := { (A → B) ∈ L_i | A ⊈ C or B ⊆ C }
14     end
15
16     i := i + 1
17   end
18
19   return K_i
20 end
```

It is the purpose of the following argumentation to show that Algorithm 5 approximately explores $\mathcal{L}$ as described above. For this we need to argue that the algorithm always terminates and, upon termination, Equation (6.2) is satisfied.

6.1.7 Proposition *Let M be a finite set, p a domain expert on M, $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and $\mathcal{K} \subseteq \mathrm{Th}(p)$. Then for iteration i of the run of* `explore-implications` *with this input, it is true that $\mathcal{K}_i \subseteq \mathcal{K}_{i+1}$ and $\mathcal{L}_i \supseteq \mathcal{L}_{i+1}$, and exactly one of these inclusions is strict.*

Proof Suppose that we are in iteration i. Then clearly $\mathcal{K}_i \subseteq \mathcal{K}_{i+1}$ and $\mathcal{L}_i \supseteq \mathcal{L}_{i+1}$ from the very definition of these sets. Then, if p confirms $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, then $\mathcal{K}_i \subsetneq \mathcal{K}_{i+1}$ and $\mathcal{L}_i = \mathcal{L}_{i+1}$. If p rejects $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, and $C = p(P_{i+1} \to \mathcal{L}_i(P_{i+1}))$, then there must exist at least one implication $(A \to B) \in \mathcal{L}_i$ such that $A \subseteq C$ and $B \nsubseteq C$, for otherwise C cannot be a counterexample for $P_{i+1} \to \mathcal{L}_i(P_{i+1})$. Therefore, $\mathcal{L}_i \supsetneq \mathcal{L}_{i+1}$, and $\mathcal{K}_i = \mathcal{K}_{i+1}$ holds by definition. □

6.1.8 Theorem *Let M be a finite set, $\leqslant_M$ a linear order on M, and let p be a domain expert on M. Let $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and $\mathcal{K} \subseteq \mathrm{Th}(p)$. Then* `explore-implications` *applied to these arguments terminates after finitely many steps.*

Proof Note that `explore-implications` has to terminate if $\mathrm{Cn}(\mathcal{L}_i)\backslash\mathrm{Cn}(\mathcal{K}_i) = \varnothing$. By Proposition 6.1.7 we know that $\mathcal{K}_i \subseteq \mathcal{K}_{i+1}$ or $\mathcal{L}_i \supseteq \mathcal{L}_{i+1}$, and exactly one of these inclusions is strict. The latter fact entails

$$\mathrm{Cn}(\mathcal{L}_{i+1})\backslash\mathrm{Cn}(\mathcal{K}_{i+1}) \subsetneq \mathrm{Cn}(\mathcal{L}_i)\backslash\mathrm{Cn}(\mathcal{K}_i). \tag{6.3}$$

This is because if $\mathcal{K}_i \neq \mathcal{K}_{i+1}$ then the implication $P_{i+1} \to \mathcal{L}_i(P_{i+1})$ added to $\mathcal{K}_i$ does not follow from $\mathcal{K}_i$, because $P_{i+1} = \mathcal{K}_i(P_{i+1})$. Thus, $\mathrm{Cn}(\mathcal{K}_i) \subsetneq \mathrm{Cn}(\mathcal{K}_{i+1})$. On the other hand, if $\mathcal{L}_i \neq \mathcal{L}_{i+1}$, then

$$(P_{i+1} \to \mathcal{L}_i(P_{i+1})) \in \mathrm{Cn}(\mathcal{L}_i)\backslash\mathrm{Cn}(\mathcal{L}_{i+1})$$

and $\mathrm{Cn}(\mathcal{L}_i) \supsetneq \mathrm{Cn}(\mathcal{L}_{i+1})$.

If the algorithm would run forever, it would thus yield an infinite descending chain

$$\mathrm{Cn}(\mathcal{L}_0)\backslash\mathrm{Cn}(\mathcal{K}_0) \supsetneq \mathrm{Cn}(\mathcal{L}_1)\backslash\mathrm{Cn}(\mathcal{K}_1) \supsetneq \dots$$

Since M is finite, $\mathfrak{P}(\mathrm{Imp}(M))$ is finite as well, and therefore this cannot happen. Thus the algorithm has to terminate. □

We now consider the correctness of Algorithm 5.

6.1.9 Proposition *Let M be a finite set, p a domain expert on M, and $\leqslant_M$ a linear order on M. Let $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and $\mathcal{K} \subseteq \mathrm{Th}(p)$. Denote with $\preceq$ the lectic order on $\mathfrak{P}(M)$ induced by $\leqslant_M$. Then in every iteration i of the run of* `explore-implications` *with this input such that $P_{i+1} \neq$ **nil**, it is true that for $A \prec P_{i+1}$, if A is $\mathcal{K}_i$-closed, then A is $\mathcal{L}_i$-closed as well.*

Proof We show the claim by induction on i. For the base case $i = 0$ the claim is vacuously true, since P_1, if not **nil**, is the lectically smallest set which is $\mathcal{K}_0$-closed but not $\mathcal{L}_i$-closed.

For the step case we assume that $P_{i+2} \neq \mathbf{nil}$. We then need to show that for all $A \prec P_{i+2}$, if A is $\mathcal{K}_{i+1}$-closed, then A is also $\mathcal{L}_{i+1}$-closed.

Thus let $A \prec P_{i+2}$ and A be $\mathcal{K}_{i+1}$-closed. Let us first consider the case $A \prec P_{i+1}$. Since $\mathcal{K}_i \subseteq \mathcal{K}_{i+1}$, A is also $\mathcal{K}_i$-closed. By induction hypothesis, A is also $\mathcal{L}_i$-closed, and since $\mathcal{L}_{i+1} \subseteq \mathcal{L}_i$, we obtain that A is $\mathcal{L}_{i+1}$-closed.

If $P_{i+1} = P_{i+2}$, nothing remains to be shown. Therefore assume that $P_{i+1} \neq P_{i+2}$ and let $P_{i+1} \preceq A \prec P_{i+2}$. By construction, all sets strictly between P_{i+1} and P_{i+2} are $\mathcal{L}_{i+1}$-closed if $\mathcal{K}_{i+2}$-closed, by definition of P_{i+2}.

Therefore, only the case $A = P_{i+1} \neq P_{i+2}$ remains to be considered. But then since A is $\mathcal{K}_{i+2}$-closed and $P_{i+1} \neq P_{i+2}$ we know that P_{i+1} is not $\mathcal{L}_{i+1}$-closed, as otherwise $P_{i+1} = P_{i+2}$. Therefore, $A = P_{i+1}$ is $\mathcal{L}_{i+1}$-closed as required. □

6.1.10 Corollary *Let M, $\leqslant_M$, p, $\mathcal{L}$, $\mathcal{K}$ as before, and denote with n the number of iterations of the algorithm* `explore-implications` *when applied to this input. Then*

$$\mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathcal{L}_n \cup \mathcal{K}).$$

Proof By the previous Proposition 6.1.9 we know that all $\mathcal{K}_n$-closed subsets of M are also $\mathcal{L}_n$-closed. Let $(X \to Y) \in \mathrm{Cn}(\mathcal{L}_n)$. Then $Y \subseteq \mathcal{L}_n(X) \subseteq \mathcal{L}_n(\mathcal{K}_n(X))$. Clearly, $\mathcal{K}_n(X)$ is $\mathcal{K}_n$-closed, so Proposition 6.1.9 yields that $\mathcal{K}_n(X)$ is also $\mathcal{L}_n$-closed, i. e.

$$\mathcal{L}_n(\mathcal{K}_n(X)) = \mathcal{K}_n(X).$$

Thus, $Y \subseteq \mathcal{K}_n(X)$ and therefore $(X \to Y) \in \mathrm{Cn}(\mathcal{K}_n)$. Clearly, $\mathcal{K} \subseteq \mathrm{Cn}(\mathcal{K}_n)$, and thus $\mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathcal{L}_n \cup \mathcal{K})$ as required. □

6.1.11 Corollary *Let M, $\leqslant_M$, p, $\mathcal{L}$, $\mathcal{K}$ as before, and let again n be the number of iterations of the run of* `explore-implications` *when applied to this input. Then*

$$\mathcal{L}_n = \mathrm{Th}(p) \cap \mathcal{L}.$$

Proof We know that $\mathrm{Th}(p) \cap \mathcal{L} \subseteq \mathcal{L}_n$, since p does not provide counterexamples for confirmed implications. Since $\mathrm{Cn}(\mathcal{L}_n \cup \mathcal{K}) \subseteq \mathrm{Cn}(\mathcal{K}_n)$ by Corollary 6.1.10, we obtain $\mathcal{L}_n \subseteq \mathrm{Cn}(\mathcal{K}_n)$. Since all implications in $\mathcal{K}_n$ are confirmed by p, it is true that $\mathrm{Cn}(\mathcal{K}_n) \subseteq \mathrm{Th}(p)$. Together with $\mathcal{L}_n \subseteq \mathcal{L}$ we thus obtain $\mathcal{L}_n \subseteq \mathrm{Th}(p) \cap \mathcal{L}$. □

6.1.12 Theorem *Let M be a finite set, $\leqslant_M$ be a linear order on M, and p a domain expert on M. Let $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and $\mathcal{K} \subseteq \mathrm{Th}(p) \cap \mathrm{Cn}(\mathcal{L})$. If n is the last iteration of* `explore-implications` *applied to this input, then*

$$\mathrm{Th}(p) \cap \mathrm{Cn}(\mathcal{L}) \supseteq \mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathrm{Th}(p) \cap \mathcal{L}).$$

Proof Since $\mathcal{K} \subseteq \mathrm{Cn}(\mathcal{L})$, it is true that $\mathcal{K} \subseteq \mathrm{Cn}(\mathcal{L}_n)$ and Corollary 6.1.10 yields $\mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathcal{L}_n)$. Thus

$$\mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathcal{L}_n) = \mathrm{Cn}(\mathrm{Th}(p) \cap \mathcal{L})$$

by Corollary 6.1.11.

Furthermore, $\mathrm{Cn}(\mathcal{L}) \supseteq \mathrm{Cn}(\mathcal{K}_n)$ holds by the definition of $\mathcal{K}_n$. Since all implications in $\mathcal{K}$ have been confirmed by the expert, $\mathrm{Th}(p) \supseteq \mathrm{Cn}(\mathcal{K}_n)$ also holds. This yields

$$\mathrm{Cn}(\mathcal{L}) \cap \mathrm{Th}(p) \supseteq \mathrm{Cn}(\mathcal{K}_n),$$

as required. □

The counterexamples given by the expert p during the run of the algorithm can be collected into a formal context $\mathbb{L}$. Then this formal context $\mathbb{L}$ has the nice property that every implication in $\mathcal{L}$ is either entailed by $\mathcal{K}_n$, or does not hold in $\mathbb{L}$. This fact can be useful on its own.

6.1.13 Theorem *Let M be a finite set, $\leqslant_M$ a linear order on M, and let p be a domain expert on M. Let $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and $\mathcal{K} \subseteq \mathrm{Th}(p) \cap \mathrm{Cn}(\mathcal{L})$. Let n be the last iteration of* `explore-implications` *applied to this input, and let $\{ C_1, \ldots, C_m \}$ be the counterexamples given by p during this run. Denote with $\mathbb{L}$ the formal context which arises from these counterexamples, i. e.*

$$\mathbb{L} = (\{ C_1, \ldots, C_m \}, M, \ni).$$

Then for each implication $(A \to B) \in \mathcal{L}$ either $(A \to B) \in \mathrm{Cn}(\mathcal{K}_n)$ or $(A \to B) \notin \mathrm{Th}(\mathbb{L})$, i. e.

$$\mathrm{Cn}(\mathcal{K}_n) \cap \mathcal{L} = \mathrm{Th}(\mathbb{L}) \cap \mathcal{L}.$$

Proof We first observe that

$$\mathcal{L}_n = \mathrm{Th}(\mathbb{L}) \cap \mathcal{L}. \tag{6.4}$$

Then from Corollary 6.1.10 we obtain

$$\mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathcal{L}_n) = \mathrm{Cn}(\mathrm{Th}(\mathbb{L}) \cap \mathcal{L}). \tag{6.5}$$

Suppose that $(A \to B) \notin \mathrm{Cn}(\mathcal{K}_n)$. Then $(A \to B) \notin \mathrm{Cn}(\mathrm{Th}(\mathbb{L}) \cap \mathcal{L})$ by Equation (6.5). Since $(A \to B) \in \mathcal{L}$, we obtain $(A \to B) \notin \mathrm{Th}(\mathbb{L})$ as required.

On the other hand, since p does not provide counterexamples for confirmed implications, and all implications in $\mathcal{K}_n$ have been confirmed by p, we can infer that $\mathrm{Cn}(\mathcal{K}_n) \subseteq \mathrm{Th}(\mathbb{L})$.□

An interesting observation for Algorithm 5 is the following: although it may not hold that $\mathcal{K}_n$ is a base of $\mathcal{L} \cap \mathrm{Th}(p)$ with background knowledge $\mathcal{K}$, the set of implications computed by `explore-implications` is *minimal* in the sense that it is the canonical base of itself, i. e.

$$\mathrm{Can}(\mathcal{K}_n, \mathcal{K}) = \mathcal{K}_n \backslash \mathcal{K}.$$

This fact is indeed not very surprising, given that we have obtained Algorithm 5 from Algorithm 4 by simple syntactic substitutions.

Note that we had introduced the canonical base in Section 2.4 only for formal contexts, but it is not very difficult to generalize the definition (and the corresponding results) to the level of sets of implications. Indeed, if $\mathcal{L}, \mathcal{K} \subseteq \mathrm{Imp}(M)$, then the canonical base of $\mathcal{L}$ with background knowledge $\mathcal{K}$ can just be defined to be the canonical base of $\mathbb{K}_{\mathcal{L}}$ with background knowledge $\mathcal{K}$, where $\mathbb{K}_{\mathcal{L}}$ is defined as in Proposition 2.3.7. However, we can also be a bit more specific and generalize the notion of pseudo-intents correspondingly, by considering the closure operator induced by $\mathcal{L}$ to be a generalization of the double-prime operator $(\cdot)''$ of a formal context.

6.1.14 Definition ($\mathcal{K}$-Pseudo-Closed Set of $\mathcal{L}$, Canonical Base) Let M be a finite set and let $\mathcal{L}, \mathcal{K} \subseteq \mathrm{Imp}(M)$. Then a set $P \subseteq M$ is called a *$\mathcal{K}$-pseudo-closed set* of $\mathcal{L}$ if and only if

i. $P \neq \mathcal{L}(P)$,

ii. $P = \mathcal{K}(P)$, and

iii. for all $Q \subsetneq P$ being a $\mathcal{K}$-pseudo-closed set of $\mathcal{L}$ it is true that $\mathcal{L}(Q) \subseteq P$.

The *canonical base* of $\mathcal{L}$ with background knowledge $\mathcal{K}$ is then defined as

$$\mathrm{Can}(\mathcal{L}, \mathcal{K}) := \{\, P \to \mathcal{L}(P) \mid P \subseteq M \text{ a } \mathcal{K} \text{ pseudo-closed set of } \mathcal{L} \,\}. \qquad \diamondsuit$$

The relevant property of the canonical base now carries over to this generalized formulation, just because

$$\mathrm{Can}(\mathcal{L}, \mathcal{K}) = \mathrm{Can}(\mathbb{K}_{\mathcal{L}}, \mathcal{K}).$$

Thus, the following corollary follows directly from Theorem 2.4.7.

6.1.15 Corollary *Let M be a finite set and let $\mathcal{L}, \mathcal{K} \subseteq \mathrm{Imp}(M)$. Then $\mathrm{Can}(\mathcal{L}, \mathcal{K})$ is a set of valid implications of $\mathcal{L}$ such that*

$$\mathrm{Can}(\mathcal{L}, \mathcal{K}) \cup \mathcal{K}$$

is complete for $\mathcal{L}$, and $\mathrm{Can}(\mathcal{L}, \mathcal{K})$ has minimal cardinality with this property. In particular, if $\mathcal{K} \subseteq \mathrm{Cn}(\mathcal{L})$, then $\mathrm{Can}(\mathcal{L}, \mathcal{K})$ is a minimal base of $\mathcal{L}$ with background knowledge $\mathcal{K}$.

We now show that Algorithm 5 computes the canonical base of $\mathrm{Th}(p) \cap \mathcal{L}$ with background knowledge $\mathcal{K}$.

6.1.16 Theorem *Let M be a finite set, $\leqslant_M$ be a linear order on M, p a domain expert on M, $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and $\mathcal{K} \subseteq \mathrm{Th}(p) \cap \mathcal{L}$. If*

$$\mathcal{K}_n = \texttt{explore-implications}(M, \leqslant_M, p, \mathcal{L}, \mathcal{K}),$$

then

$$\mathrm{Can}(\mathcal{K}_n, \mathcal{K}) = \mathcal{K}_n \backslash \mathcal{K}.$$

Proof First note that $\mathcal{K}_n \backslash \mathcal{K}$ is an irredundant base of $\mathcal{K}_n$ with background knowledge $\mathcal{K}$, i. e. no implication in $\mathcal{K}_n$ follows from the others and $\mathcal{K}$.

We are going to show that the premises in $\mathcal{K}_n \backslash \mathcal{K}$ are all $\mathcal{K}$-pseudo-closed sets of $\mathcal{K}_n$. For this, we show the following claim by induction over the number i of iterations:

> For every iteration i, the $k := |\mathcal{K}_n \backslash \mathcal{K}|$ lectically first $\mathcal{K}$-pseudo-closed sets of $\mathcal{K}_n$ are precisely the premises of the implications in $\mathcal{K}_n \backslash \mathcal{K}$. If the $(k+1)$st $\mathcal{K}$-pseudo-closed set Q of $\mathcal{K}_n$ exists, then $P_{i+1} \subseteq M$ (i. e. is not **nil**), and $P_{i+1} \preceq Q$.

For the base case $i = 0$ we observe that $|\mathcal{K}_0 \backslash \mathcal{K}| = 0$, thus the first part of the claim holds. If Q is the first $\mathcal{K}$-pseudo-closed set of $\mathcal{K}_n$, then Q is $\mathcal{K}$-closed but not $\mathcal{K}_n$-closed. But then Q is also not $\mathcal{L}$-closed, because $\mathcal{K}_n \subseteq \mathrm{Cn}(\mathcal{L})$. Therefore, by construction, P_1 is not **nil** and $P_1 \preceq Q$.

For the induction step we assume that the claim holds for iteration i. Let $k := |\mathcal{K}_i \backslash \mathcal{K}|$. If there is no more $\mathcal{K}$-pseudo-closed set of $\mathcal{K}_n$ which is not already a premise of an implication in $\mathcal{K}_i \backslash \mathcal{K}$, then $\mathcal{K}_i$ is a base of $\mathcal{K}_n$ with background knowledge $\mathcal{K}$. Since $\mathcal{K}_i \subseteq \mathcal{K}_n$ we obtain $\mathcal{K}_i = \mathcal{K}_n$, since $\mathcal{K}_n$ is irredundant. Then $i = n$, or $i < n$ and $|\mathcal{K}_{i+1} \backslash \mathcal{K}_n| = k$ and the claim holds for iteration $i + 1$ as well.

Now assume that Q is the $(k+1)$st $\mathcal{K}$-pseudo-closed set of $\mathcal{K}_n$. By induction hypothesis, $P_{i+1} \subseteq M$ and $P_{i+1} \preceq Q$. We consider two main cases.

Case $P_{i+1} = Q$: If $\mathcal{L}_i(P_{i+1}) \subseteq \mathcal{K}_n(P_{i+1})$, then p accepts the implication

$$P_{i+1} \to \mathcal{L}_i(P_{i+1}),$$

which is then an element of $\mathcal{K}_{i+1}$. Thus, the $|\mathcal{K}_{i+1} \backslash \mathcal{K}| = k + 1$ lectically first $\mathcal{K}$-pseudo-closed sets of $\mathcal{K}_n$ are precisely the premises of the implications in $\mathcal{K}_{i+1} \backslash \mathcal{K}$.

If the $(k+2)$nd $\mathcal{K}$-pseudo-closed set $\overline{Q}$ of $\mathcal{K}_n$ exists, then in particular $\overline{Q}$ is closed under $\mathcal{K}_{i+1}$, as for each $\mathcal{K}$-pseudo-closed set $P_\ell \subseteq \overline{Q}$ it is true tat $\mathcal{K}_n(P_\ell) \subseteq \overline{Q}$. Furthermore, $\overline{Q}$ is not $\mathcal{K}_n$-closed, and thus also not $\mathcal{L}_{i+1}$-closed. Therefore, by construction $P_{i+2} \subseteq M$, i. e. is not **nil**, and $P_{i+2} \preceq Q$.

If $\mathcal{L}_i(P_{i+1}) \not\subseteq \mathcal{K}_n(P_{i+1})$, then p rejects the implication $P_{i+1} \to \mathcal{L}_i(P_{i+1})$ and provides a counterexample. Then $\mathcal{K}_{i+1} = \mathcal{K}_i$. Since $P_{i+1} = Q$, the set P_{i+1} is also not $\mathcal{L}_{i+1}$-closed, as otherwise it would be $\mathcal{K}_n$-closed. Thus, $P_{i+2} = P_{i+1}$ and $P_{i+2} \preceq Q$ as required.

Case $P_{bi+1} \precneqq Q$: In this case, the set P_{i+1} must be $\mathcal{K}_n$-closed, since otherwise it would be a $\mathcal{K}$-pseudo-closed set of $\mathcal{K}_n$, and $P_{i+1} \succeq Q$ would hold. To see this, we first observe that P_{i+1} is K_i-closed by definition, and thus also $\mathcal{K}$-closed. Furthermore, if $\overline{Q} \subsetneq P_{i+1}$ is a $\mathcal{K}$-pseudo-closed set of $\mathcal{K}_n$, then by induction hypothesis, the set $\overline{Q}$ is a premise of $\mathcal{K}_i$ and therefore $\mathcal{K}_n(\overline{Q}) \subseteq P_{i+1}$. Thus, the only possibility for P_{i+1} not to be a $\mathcal{K}$-pseudo-closed set of $\mathcal{K}_n$ is that P_{i+1} is $\mathcal{K}_n$-closed.

In addition, P_{i+1} is not $\mathcal{L}_i$-closed by definition. Thus, since P_{i+1} is $\mathcal{K}_n$-closed, the expert rejects the implication $P_{i+1} \to \mathcal{L}_i(P_{i+1})$. Therefore, $\mathcal{K}_{i+1} = \mathcal{K}_i$ and P_{i+2} is the lectically smallest $\mathcal{K}_{i+1}$-closed, not $\mathcal{L}_{i+1}$-closed set which is lectically greater or equal to P_{i+1}. Since Q is also $\mathcal{K}_{i+1}$-closed (by induction hypothesis, as it is a $\mathcal{K}$-pseudo-closed set of $\mathcal{K}_n$) but not $\mathcal{L}_{i+1}$-closed (since Q is not $\mathcal{K}_n$-closed), it is true that $P_{i+2} \preceq Q$ by construction.

This finishes the step case and the proof of the above claim. Using this claim, we shall now show the theorem. Note that for each implication $(P_i \to \mathcal{L}_{i-1}(P_i)) \in \mathcal{K}_n \backslash \mathcal{K}$ it is true that

$$\mathcal{L}_{i-1}(P_i) = \mathcal{K}_n(P_i) \tag{6.6}$$

because $\mathcal{L}_i(P_i)$ is already $\mathcal{K}_n$-closed, and $(P_i \to \mathcal{L}_{i-1}(P_i)) \in \mathcal{K}_n \backslash \mathcal{K}$. If now n denotes the last iteration of the algorithm, we known that P_{n+1} is equal to **nil**. Therefore, there cannot exist a $\mathcal{K}$-pseudo-closed set of $\mathcal{K}_n$ which is not a premise of an implication in $\mathcal{K}_n \backslash \mathcal{K}$. Because of Equation (6.6), we therefore obtain

$$\mathcal{K}_n \backslash \mathcal{K} = \mathrm{Can}(\mathcal{K}_n, \mathcal{K})$$

as required. □

Recall that our initial motivation to consider the generalization of attribute exploration as given in Algorithm 5 was to be able to explore sets of the form $\mathrm{Th}_c(\mathbb{K})$ for arbitrary choices of $c \in [0, 1]$. However, for this particular application our algorithm does not seem very practical, as during a run we would need to compute sets of the form $\mathcal{L}_i(P)$ for some $\mathcal{L}_i \subseteq \mathrm{Th}_c(\mathbb{K})$ and $P \subseteq M$. For $c = 1$ this is not a problem, since

$$\mathrm{Th}_1(\mathbb{K})(P) = \mathrm{Th}(\mathbb{K})(P) = P''.$$

For $c \neq 1$, however, such a simple computation is not known. Although we shall discuss in Section 6.2.1 some techniques to still achieve the computation of $\mathrm{Th}_c(\mathbb{K})(P)$, the computation itself is potentially much more expensive than in the case $c = 1$.

Also, the fact that Algorithm 5 only provides an approximate exploration of $\mathcal{L}$ using p as an expert is unsatisfactory. Of course, this situation can easily be avoided if the

implications asked to the expert are elements of $\mathcal{L}$, because then $\mathrm{Cn}(\mathcal{K}_n) \subseteq \mathcal{L} \cap \mathrm{Th}(p)$ and thus Theorem 6.1.12 yields that

$$\mathrm{Cn}(\mathcal{K}_n) = \mathrm{Cn}(\mathcal{L} \cap \mathrm{Th}(p)),$$

i. e. $\mathcal{K}_n$ is a base of $\mathcal{L} \cap \mathrm{Th}(p)$.

However, it cannot be guaranteed in general that the implications $P_{i+1} \to \mathcal{L}_i(P_{i+1})$ are elements of $\mathcal{L}$. To remedy this problem, we shall discuss in the following a relaxation of Algorithm 5 that allows for more freedom in the choice which implications are asked to the expert. In this way, it may be easier to ensure that the implications asked are indeed elements of $\mathcal{L}$, and thus the exploration does not contain too many questions. We shall see in Section 6.2.2 how this idea can be applied for our case of exploration by confidence.

In this relaxation, instead of asking questions of the form $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, it will be sufficient to find *some* $Q \subseteq M$ such that $P_{i+1} \subsetneq Q \subseteq \mathcal{L}_i(P_{i+1})$, and then ask the implication $P \to Q$ to the expert. Of course, we have to pay for the freedom of choosing the set Q freely by potentially asking more questions, since the implication

$$P_{i+1} \to \mathcal{L}_i(P_{i+1}) \backslash Q$$

still needs to be examined (implicitly or explicitly) by the expert. The hope is that the freedom of leaving out elements from $\mathcal{L}_i(P_{i+1})$ compensates for this extra amount of implications the expert has to handle.

Algorithm 6 presents a generalization of Algorithm 5 that uses these ideas. Algorithm 6 is very similar to Algorithm 5, but differs in a crucial aspect: we still compute sets of the form

$$P = \texttt{next-closed-non-closed}(M, \leqslant_M, P_i, \mathcal{K}_i, \mathcal{L}_i),$$

but we then do not directly ask $P \to \mathcal{L}_i(P)$ to the expert. Instead, we allow the algorithm to choose a set P_{i+1} such that

$$P_i \preceq P_{i+1} \preceq P$$

and P_{i+1} is not $\mathcal{L}_i$-closed. Note that such a set always exists since P is not $\mathcal{L}_i$-closed.

Since P_{i+1} is not $\mathcal{L}_i$-closed, $P_{i+1} \subsetneq \mathcal{L}_i(P_{i+1})$. We then choose a set Q such that

$$P_{i+1} \subsetneq Q \subseteq \mathcal{L}_i(P_{i+1})$$

and finally ask the implication $P \to Q$ to the expert.

Note that if we always choose $P_{i+1} = P$ and $Q = \mathcal{L}_i(P_{i+1})$, then we obtain Algorithm 5 again.

We now argue that Algorithm 6 is indeed an algorithm that allows to approximately explore $\mathcal{L}$, using p as an expert and $\mathcal{K}$ as background knowledge. For this we first observe that termination can be argued as in the case of Algorithm 5, and therefore the analog of Theorem 6.1.8 holds.

Algorithm 6 (Generalized Exploration of Sets of Implications)

```
0  define explore-implications/weaker-version(M, ⩽_M, p, L ⊆ Imp(M), K ⊆ Th(p))
1    i := 0, L_i := L, K_i := K, P_i := ∅
2
3    forever do
4      P := next-closed-non-closed(M, ⩽_M, P_i, K_i, L_i)
5      if P = nil exit
6      choose P_{i+1} ⊆ M such that P_i ≺ P_{i+1} ≺ P and P_{i+1} not L_i-closed
7      choose Q ⊆ M such that P_{i+1} ⊊ Q ⊆ L_i(P_{i+1}), Q ⊈ K_i(P_{i+1})
8
9      if p(P_{i+1} → Q) = ⊤ then
10       K_{i+1} := K_i ∪ { P_{i+1} → Q }
11       L_{i+1} := L_i
12     else
13       K_{i+1} := K_i
14       C := p(P_{i+1} → Q)
15       L_{i+1} := { (A → B) ∈ L_i | A ⊈ C or B ⊆ C }
16     end
17
18     i := i + 1
19   end
20
21   return K_i
22 end
```

6.1.17 Theorem *Let M be a finite set, $\leqslant_M$ a linear order on M, p be a domain expert on M, $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and $\mathcal{K} \subseteq \mathrm{Th}(p)$. Then the call to Algorithm 6 with this input terminates after finitely many steps.*

For the correctness of Algorithm 6 we can argue in practically the same way as we did for Algorithm 5. The crucial argument there was Proposition 6.1.9, which also holds in this case, with nearly the same proof. We shall repeat it nevertheless for easier comparison.

6.1.18 Proposition *Let M, $\leqslant_M$, p, $\mathcal{L}$, $\mathcal{K}$ as before. Then for every iteration i in the run of*

`explore-implications/weaker-version`$(M, \leqslant_M, p, \mathcal{L}, \mathcal{K})$

*if P_{i+1} is not **nil**, it is true for all $A \prec P_{i+1}$ that, if A is $\mathcal{K}_i$-closed, then A is also $\mathcal{L}_i$-closed.*

Proof Again we show the claim by induction over i. For the base case $i = 0$ we know that $P_i = \varnothing$, and thus the claim is vacuously true.

For the step case assume the validity of the proposition for iteration i. Then if P_{i+2} is not **nil** we need to show that for each $A \preceq P_{i+2}$, if A is $\mathcal{K}_{i+1}$-closed, then A is also $\mathcal{L}_{i+1}$-closed.

As before we can reduce this to the case that $P_{i+1} \preceq A \precnsim P_{i+2}$ and A being $\mathcal{K}_{i+1}$-closed. In particular $P_{i+1} \neq P_{i+2}$. Then $P_{i+1} = A$ is the only interesting case, since all sets strictly lectically lying between P_{i+1} and P_{i+2} which are $\mathcal{K}_{i+1}$-closed are also $\mathcal{L}_{i+1}$-closed, due to the construction of P_{i+2}. Since $P_{i+1} \neq P_{i+2}$ we know that P_{i+1} must also be $\mathcal{L}_{i+1}$-closed, as otherwise the algorithm would compute $P_{i+1} = P_{i+2}$. Thus, $A = P_{i+1}$ is also $\mathcal{L}_{i+1}$-closed, as required. □

Now most of the properties of Algorithm 5 carry over to Algorithm 6. However, since we do not necessarily ask implications of the form

$$P_{i+1} \to \mathcal{L}_i(P_{i+1}) \tag{6.7}$$

anymore, we can also not expect that the algorithm computes the canonical base of $\mathcal{K}_n$ with background knowledge $\mathcal{K}$. In particular this means that the number of implications asked to and confirmed by the expert is not necessarily minimal.

6.1.19 Theorem *Let M be a finite set, $\leqslant_M$ be a linear order on M, and let p be a domain expert on M. Let $\mathcal{L} \subseteq \mathrm{Imp}(M)$ and $\mathcal{K} \subseteq \mathrm{Th}(p) \cap \mathrm{Cn}(\mathcal{L})$. Denote with n the last iteration of the run of Algorithm 6 applied to this input. Then*

i. $\mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathcal{L}_n)$,

ii. $\mathcal{L}_n = \mathrm{Th}(p) \cap \mathcal{L}$, *and*

iii. $\mathrm{Th}(p) \cap \mathrm{Cn}(\mathcal{L}) \supseteq \mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathrm{Th}(p) \cap \mathcal{L})$.

6.2. Exploration by Confidence

As a first step to achieve an algorithm for exploration by confidence we shall instantiate Algorithm 5 for $\mathcal{L} = \mathrm{Th}_c(\mathbb{K})$. This will be done in Section 6.2.1. While this approach is conceptually trivial, it entails some technical problems, the most serious of them being able to compute closures of the form

$$\mathrm{Th}_c(\mathbb{K})(P)$$

for sets $P \subseteq M$. We discuss some approaches how to achieve this computation more efficiently than by iterating through $\mathrm{Th}_c(\mathbb{K})$, which are, however, still potentially expensive. Furthermore, in the general case this exploration will only be approximative, which may or may not be a problem, depending on the current application.

To remedy these problems, we use our weaker formulation of Algorithm 6 and derive from it in Section 6.2.2 an algorithm for exploration by confidence that avoids computing closures under $\mathrm{Th}_c(\mathbb{K})$. In addition, for this algorithm we can ensure that all implications asked to the expert satisfy the confidence constraint given by c. The price we have to pay for this is an increase in the number of questions asked to the expert, and the fact that we do not compute the canonical base anymore.

6.2.1. An Approximative Exploration by Confidence

We obtain our first algorithm for exploration by confidence by instantiating Algorithm 5 with $\mathcal{L} = \mathrm{Th}_c(\mathbb{K})$. The first problem we have to face here is how to implement Line 13 in the algorithm, i. e.

$$\mathcal{L}_{i+1} := \{\, (A \to B) \in \mathcal{L}_i \mid A \not\subseteq C \text{ or } B \subseteq C \,\}.$$

Naively, if $\mathcal{L} = \mathrm{Th}_c(\mathbb{K})$, this statement can be implemented by enumerating all implications in $\mathrm{Th}_c(\mathbb{K})$, an approach which we want to avoid for obvious reasons. Instead, we collect all counterexamples in a formal context $\mathbb{L}_i$ as we did in Theorem 6.1.13, and then use the fact which we already used in the proof of this statement (Equation (6.4)), which in our case amounts to

$$\mathcal{L}_i = \mathrm{Th}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i).$$

The algorithm we thus obtain is shown in Algorithm 7. The following result is an immediate consequence of the results we have obtained for Algorithm 5.

6.2.1 Corollary *Let $\mathbb{K} = (G, M, I)$ be a finite formal context, $\leqslant_M$ a linear order on M, $c \in [0, 1]$, p a domain expert on M and $\mathcal{K} \subseteq \mathrm{Th}_c(\mathbb{K}) \cap \mathrm{Th}(p)$. Then the call of Algorithm 7 with this input terminates after finitely many steps. If n is the number of iterations of this call, and if $\mathcal{K}_n$ is the corresponding return value, then*

$$\mathrm{Th}(p) \cap \mathrm{Cn}(\mathrm{Th}_c(\mathbb{K})) \supseteq \mathrm{Cn}(\mathcal{K}_n) \supseteq \mathrm{Cn}(\mathrm{Th}(p) \cap \mathrm{Th}_c(\mathbb{K})).$$

Algorithm 7 (Exploration by Confidence, Approximative Version)

```
0  define exploration-by-confidence(𝕂 = (G, M, I), ⩽_M, p, c ∈ [0,1], 𝒦 ⊆ Th(p))
1    i := 0, 𝕃_i = (∅, M, ∅), 𝒦_i := 𝒦, P_i := ∅
2
3    forever do
4      ℒ_i := Th_c(𝕂) ∩ Th(𝕃_i)
5      P_{i+1} := next-closed-non-closed(M, ⩽_M, P_i, 𝒦_i, ℒ_i)
6      if P_{i+1} = nil exit
7
8      if p(P_{i+1} → ℒ_i(P_{i+1})) = ⊤ then
9        𝒦_{i+1} := 𝒦_i ∪ { P_{i+1} → ℒ_i(P_{i+1}) }
10       𝕃_{i+1} := 𝕃_i
11     else
12       𝒦_{i+1} := 𝒦_i
13       𝕃_{i+1} := 𝕃_i + p(P_{i+1} → ℒ_i(P_{i+1})) ;; add counterexample to 𝕃_i
14     end
15
16     i := i + 1
17   end
18
19   return 𝒦_i
20 end
```

Moreover, $\mathcal{K}_n \backslash \mathcal{K}$ is the canonical base of itself.

Recall that when we motivated exploration by confidence we mentioned the need to distinguish between different forms of counterexamples: if $(A \to B) \in \mathrm{Imp}(M)$, then the counterexamples for $A \to B$ contained in $\mathbb{K}$ can possibly be ignored if only $\mathrm{conf}_{\mathbb{K}}(A \to B) \geqslant c$. On the other hand, if our expert p provides a counterexample for $A \to B$, then this counterexample cannot be ignored, even if the confidence of $A \to B$ would still be sufficiently high.

Our Algorithm 7 solves this problem by just considering two formal contexts, namely $\mathbb{K}$ on the one hand, where we apply the confidence threshold, and $\mathbb{L}_i$ on the other hand, of which we only consider valid implications. This distinction is immediate if we consider the definition of $\mathcal{L}_i$, namely

$$\mathcal{L}_i = \mathrm{Th}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i).$$

Of course this solution is only practical if we can efficiently compute closures under $\mathcal{L}_i$. As already mentioned, we want to avoid enumerating $\mathrm{Th}_c(\mathbb{K})$ at all costs, as otherwise the algorithm becomes impractical even from a theoretical point of view.

It is worth noting that an apparently simpler approach to avoid computing closures under $\mathrm{Th}_c(\mathbb{K})$ does not work. More precisely, one may be tempted to replace in Algorithm 7 all implications of the form

$$P_{i+1} \to \mathcal{L}_i(P_{i+1}) \tag{6.8}$$

simply by

$$P_{i+1} \to \{\, m \in (P_{i+1})''_{\mathbb{L}_i} \mid \mathrm{conf}_{\mathbb{K}}(P_{i+1} \to \{\, m \,\}) \geqslant c \,\}. \tag{6.9}$$

However, the resulting algorithm would not be complete, i. e. implications with high confidence in $\mathbb{K}$ may not be asked to the expert. This is illustrated by the following example.

6.2.2 Example Consider the formal context $\mathbb{K}$ as given in Figure 6.1, let $\mathcal{K} = \{\, \{\, \mathsf{a} \,\} \to \{\, \mathsf{b} \,\} \,\}$, and choose $c = \frac{1}{2}$. Suppose that we apply exploration by confidence in the simplified version as described before, i. e. we ask implications of the form of (6.9) instead of those in (6.8). Then since all sets P_{i+1} are closed under $\mathcal{K}$, the implication $\{\, \mathsf{a} \,\} \to \{\, \mathsf{c} \,\}$ is never asked to the expert, because $\{\, \mathsf{a} \,\}$ is not closed under $\mathcal{K}$. On the other hand,

$$\mathrm{conf}_{\mathbb{K}}(\{\, \mathsf{a} \,\} \to \{\, \mathsf{c} \,\}) = \frac{4}{7} > \frac{1}{2},$$

i. e. $(\{\, \mathsf{a} \,\} \to \{\, \mathsf{c} \,\}) \in \mathrm{Th}_c(\mathbb{K})$, and thus should actually be asked to the expert. Furthermore, the implication $\{\, \mathsf{a} \,\} \to \{\, \mathsf{c} \,\}$ also does not follow form other implications asked to the

$\mathbb{K}$	a	b	c
1	×	×	
2	×	×	
3	×	×	
4	×	×	×
5	×	×	×
6	×		×
7	×		×
8			
9			
10			

Figure 6.1.: Context which shows that a simple approach to exploration by confidence does not work

expert, as the implications $\{\,\mathsf{b}\,\} \to \{\,\mathsf{c}\,\}$, $\{\,\mathsf{a},\mathsf{b}\,\} \to \{\,\mathsf{c}\,\}$, and $\varnothing \to \{\,\mathsf{c}\,\}$ will also not be asked to the expert, because

$$\begin{aligned}\operatorname{conf}_{\mathbb{K}}(\{\,\mathsf{b}\,\} \to \{\,\mathsf{c}\,\}) &= \frac{2}{5} < \frac{1}{2}\\ \operatorname{conf}_{\mathbb{K}}(\{\,\mathsf{a},\mathsf{b}\,\} \to \{\,\mathsf{c}\,\}) &= \frac{2}{5} < \frac{1}{2}\\ \operatorname{conf}_{\mathbb{K}}(\varnothing \to \{\,\mathsf{c}\,\}) &= \frac{4}{10} < \frac{1}{2}\end{aligned}$$

Thus, if we assume that the expert p confirms all proposed implications, and if we denote the set of confirmed implications by $\mathcal{K}_n$, then

$$\mathcal{K}_n(\{\,\mathsf{a}\,\}) = \{\,\mathsf{a},\mathsf{b}\,\}.$$

However, $\operatorname{Th}_c(\mathbb{K}) \cap \operatorname{Th}(p) = \operatorname{Th}_c(\mathbb{K})$, and

$$\operatorname{Th}_c(\mathbb{K})(\{\,\mathsf{a}\,\}) = \{\,\mathsf{a},\mathsf{b},\mathsf{c}\,\}.$$

Thus, the set $\mathcal{K}_n$ is not complete for $\operatorname{Th}_c(\mathbb{K}) \cap \operatorname{Th}(p)$. ♢

On the other hand, it is possible to ask implications as in (6.9) instead of those in (6.8) and still obtain a correct algorithm for exploration by confidence. However, to ensure completeness, extra steps are necessary in the algorithm. We shall see in Section 6.2.2 how this can be done.

In the rest of this section we shall discuss an approach which allows to compute closures under $\operatorname{Th}_c(\mathbb{K})$ at least a bit more efficiently. In what follows we shall consider derivation

in various contexts, namely in $\mathbb{K}$, $\mathbb{L}_i$ and the *subposition* of $\mathbb{K} = (G, M, I)$ and $\mathbb{L}_i = (G_i, M, I_i)$, which is the formal context

$$\mathbb{K} \div \mathbb{L}_i := \frac{\mathbb{K}}{\mathbb{L}_i} := (G \cup G_i, M, I \cup I_i),$$

where we assume that G and G_i are disjoint. To avoid confusion we shall denote the derivation operators by $(\cdot)'_{\mathbb{K}}$, $(\cdot)'_{\mathbb{L}_i}$ and $(\cdot)'_{\mathbb{K}\div\mathbb{L}_i}$, respectively. To ease readability, we shall also write $(\cdot)''_{\mathbb{K}}$ instead of $((\cdot)'_{\mathbb{K}})'_{\mathbb{K}}$, and likewise for the other contexts.

To achieve the computation of $\mathcal{L}_i(P)$ for some P we first recall the definition of $\mathcal{L}_i(P)$ as given in Definition 2.3.5. There we had defined

$$\begin{aligned}
\mathcal{L}^1(P) &:= P \cup \bigcup \{ B \mid (A \to B) \in \mathcal{L}, A \subseteq P \}, \\
\mathcal{L}^{j+1}(P) &:= \mathcal{L}^j(\mathcal{L}^1(P)) \quad (j \in \mathbb{N}_{>0}), \\
\mathcal{L}(P) &:= \bigcup_{i \in \mathbb{N}_{>0}} \mathcal{L}^i(P).
\end{aligned}$$

Therefore, if we can find a way to compute $\mathcal{L}_i^1(P)$ for $\mathcal{L}_i = \mathrm{Th}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i)$, then we are in principle able to compute $\mathcal{L}_i(P)$. Note that we only have to consider $\mathcal{L}_i^j(P)$ for up to $j = |M|$ at most, because $|\mathcal{L}_i(P)| \leqslant |M|$.

As a first observation, to compute $\mathcal{L}_i^1(P)$ for arbitrary $P \subseteq M$ we observe that

$$P''_{\mathbb{K}\div\mathbb{L}_i} \subseteq \mathcal{L}_i^1(P). \tag{6.10}$$

The reason for that is that we have $P''_{\mathbb{K}\div\mathbb{L}_i} \subseteq P''_{\mathbb{K}}$ and thus the implication $P \to P''_{\mathbb{K}\div\mathbb{L}_i}$ is valid in $\mathbb{K}$, and in particular $(P \to P''_{\mathbb{K}\div\mathbb{L}_i}) \in \mathrm{Th}_c(\mathbb{K})$. The same argumentation shows that $P''_{\mathbb{K}\div\mathbb{L}_i} \subseteq P''_{\mathbb{L}_i}$, and thus $(P \to P''_{\mathbb{K}\div\mathbb{L}_i}) \in \mathrm{Th}(\mathbb{L}_i)$. Putting these facts together we obtain

$$(P \to P''_{\mathbb{K}\div\mathbb{L}_i}) \in \mathrm{Th}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i) = \mathcal{L}_i$$

and therefore $P''_{\mathbb{K}\div\mathbb{L}_i} \subseteq \mathcal{L}_i^1(P)$.

To completely compute $\mathcal{L}_i^1(P)$ we need to consider all implications $(A \to B) \in \mathcal{L}_i$ such that $A \subseteq P$. If $A \to B$ is valid in $\mathbb{K}$, then we know that $B \subseteq P''_{\mathbb{K}\div\mathbb{L}_i}$. Therefore, by Equation (6.10), the difficult part in computing $\mathcal{L}_i^1(P)$ is to compute the set

$$\mathcal{L}_i^1(P) \setminus P''_{\mathbb{K}\div\mathbb{L}_i}.$$

One way to achieve this is to consider all implications $A \to B$ which are not valid in $\mathbb{K}$, but whose confidence is at least c in $\mathbb{K}$, i. e.

$$1 > \mathrm{conf}_{\mathbb{K}}(A \to B) \geqslant c.$$

Additionally, we can assume without loss of generality that $|B| = 1$, because

$$\mathrm{conf}_{\mathbb{K}}(A \to \{b\}) \geqslant \mathrm{conf}_{\mathbb{K}}(A \to B)$$

is true for all $b \in B$. Finally, since $\mathcal{L}_i \subseteq \mathrm{Th}(\mathbb{L}_i)$, we also know that

$$\mathcal{L}_i(P) \subseteq P''_{\mathbb{L}_i}.$$

Thus, to obtain all elements in $\mathcal{L}_i(P) \setminus P''_{\mathbb{K} \div \mathbb{L}_i}$, we can check for all elements $b \in P''_{\mathbb{L}_i} \setminus P_{\mathbb{K} \div \mathbb{L}_i}$ whether there exists a subset $A \subseteq P$ such that

$$1 > \mathrm{conf}_{\mathbb{K}}(A \to \{b\}) \geqslant c.$$

Then $y \in \mathcal{L}_i(P) \setminus P''_{\mathbb{K} \div \mathbb{L}_i}$ if and only if such a set A exists.

6.2.3 Proposition *Let $\mathbb{K} = (G, M, I)$ and $\mathbb{L}_i = (G_i, M, I)$ be two finite formal contexts such that G and G_i are disjoint. Let $c \in [0, 1]$ and define*

$$\mathcal{L}_i = \mathrm{Th}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i).$$

Then for $P \subseteq M$ it is true that

$$\mathcal{L}_i^1(P) = P''_{\mathbb{K} \div \mathbb{L}_i} \cup \{\, b \in P''_{\mathbb{L}_i} \mid \exists A \subseteq P \colon b \in A''_{\mathbb{L}_i} \setminus P''_{\mathbb{K} \div \mathbb{L}_i} \text{ and } \mathrm{conf}_{\mathbb{K}}(A \to \{b\}) \geqslant c \,\}.$$

Finding sets A as in the previous equation may be quite expensive, as in the worst case we need to search through all subsets of P. A reduction of the number of sets we have to consider is thus desirable. For this we can reuse our previous observation that

$$\mathrm{conf}_{\mathbb{K}}(A \to \{b\}) = \mathrm{conf}_{\mathbb{K}}(A''_{\mathbb{K}} \to \{b\}).$$

The idea is now that we use this fact to avoid considering all subsets of P, and just consider only those subsets which are intents of $\mathbb{K} \div \mathbb{L}_i$. Note that since $A \subseteq A''_{\mathbb{K} \div \mathbb{L}_i} \subseteq A''_{\mathbb{K}}$ we also have that

$$\mathrm{conf}_{\mathbb{K}}(A \to \{b\}) = \mathrm{conf}_{\mathbb{K}}(A''_{\mathbb{K} \div \mathbb{L}_i} \to \{b\}).$$

To enumerate all intents of $\mathbb{K} \div \mathbb{L}_i$ which are subsets of P we could use the Next Closure algorithm, as discussed in Section 2.4. However, for this we would need that $A \mapsto A''_{\mathbb{K} \div \mathbb{L}_i}$ is indeed a closure operator on P, i. e. we have to guarantee that $P = P''_{\mathbb{K} \div \mathbb{L}_i}$ is true. While this is not true in general, we can easily remedy this issue by computing $\mathcal{L}_i^1(P''_{\mathbb{K} \div \mathbb{L}_i})$ instead of $\mathcal{L}_i^1(P)$.

6.2.4 Proposition *Let $\mathbb{K} = (G, M, I)$ and $\mathbb{L}_i = (G, M, I)$ be two finite formal contexts such that G and G_i are disjoint. Let $c \in [0, 1]$ and define*

$$\mathcal{L}_i = \mathrm{Th}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i).$$

Define for $P \subseteq M$

$$\begin{aligned}\mathcal{L}_i^{1,\mathrm{conf}}(P) &:= P''_{\mathbb{K}\div\mathbb{L}_i} \cup \{\, b \in P''_{\mathbb{L}_i} \mid \exists A \subseteq P''_{\mathbb{K}\div\mathbb{L}_i} \colon A = A''_{\mathbb{K}\div\mathbb{L}_i}, b \in A''_{\mathbb{L}_i}\backslash P''_{\mathbb{K}\div\mathbb{L}_i} \\ &\qquad\qquad \text{and } \mathrm{conf}(A \to \{\, b \,\}) \geqslant c \,\}. \\ \mathcal{L}_i^{j+1,\mathrm{conf}}(P) &:= \mathcal{L}_i^{j,\mathrm{conf}}(\mathcal{L}_i^{1,\mathrm{conf}}(P)) \qquad (j \in \mathbb{N}_{>0}), \\ \mathcal{L}_i^{\mathrm{conf}}(P) &:= \bigcup_{j\in\mathbb{N}_{>0}} \mathcal{L}_i^{j,\mathrm{conf}}(P).\end{aligned}$$

Then

$$\mathcal{L}_i^{1,\mathrm{conf}}(P) = \mathcal{L}_i^1(P''_{\mathbb{K}\div\mathbb{L}_i})$$

is true for all $P \subseteq M$. In particular, $\mathcal{L}_i^{\mathrm{conf}}(P) = \mathcal{L}_i(P)$.

Note that the main benefit of $\mathcal{L}_i^{\mathrm{conf}}(P)$ over $\mathcal{L}_i(P)$ is that in the former we are allowed to only consider intents of $\mathbb{K} \div \mathbb{L}_i$ instead of all subsets of $P''_{\mathbb{K}\div\mathbb{L}_i}$. This can be done using the Next Closure algorithm as described above.

Proof Let $P \subseteq M$, and let $b \in \mathcal{L}_i^{1,\mathrm{conf}}(P)\backslash P''_{\mathbb{K}\div\mathbb{L}_i}$. Then there exists a set $A \subseteq P''_{\mathbb{K}\div\mathbb{L}_i}$ such that $A = A''_{\mathbb{K}\div\mathbb{L}_i}$, $b \in A_{\mathbb{L}_i}\backslash P''_{\mathbb{K}\div\mathbb{L}_i}$ and $\mathrm{conf}(A \to \{\, b \,\}) \geqslant c$. Since $(P''_{\mathbb{K}\div\mathbb{L}_i})''_{\mathbb{L}_i} = P''_{\mathbb{L}_i}$, the same set A shows that $b \in \mathcal{L}_i^1(P''_{\mathbb{K}\div\mathbb{L}_i})\backslash P''_{\mathbb{K}\div\mathbb{L}_i}$.

Conversely, let $b \in \mathcal{L}_i^1(P''_{\mathbb{K}\div\mathbb{L}_i})\backslash P''_{\mathbb{K}\div\mathbb{L}_i}$. Then $b \in (P''_{\mathbb{K}\div\mathbb{L}_i})''_{\mathbb{L}_i} = P''_{\mathbb{L}_i}$. Furthermore, there exists a set $A \subseteq P''_{\mathbb{K}\div\mathbb{L}_i}$ such that $b \in A''_{\mathbb{L}_i}\backslash P''_{\mathbb{K}\div\mathbb{L}_i}$ and $\mathrm{conf}_{\mathbb{K}}(A \to \{\, b \,\}) \geqslant c$. Then $\overline{A} := A''_{\mathbb{K}\div\mathbb{L}_i}$ satisfies $\overline{A} = \overline{A}''_{\mathbb{K}\div\mathbb{L}_i}$, $b \in \overline{A}''_{\mathbb{L}_i}\backslash P''_{\mathbb{K}\div\mathbb{L}_i}$ and $\mathrm{conf}_{\mathbb{K}}(\overline{A} \to \{\, b \,\}) \geqslant c$, thus $b \in \mathcal{L}_i^{1,\mathrm{conf}}(P)\backslash P''_{\mathbb{K}\div\mathbb{L}_i}$.

Finally, because of $\mathcal{L}_i^{1,\mathrm{conf}}(P) = \mathcal{L}_i^1(P''_{\mathbb{K}\div\mathbb{L}_i})$, we know that

$$\mathcal{L}_i^1(P) \subseteq \mathcal{L}_i^{1,\mathrm{conf}}(P) \subseteq \mathcal{L}_i(P),$$

and therefore $\mathcal{L}_i^{\mathrm{conf}}(P) = \mathcal{L}_i(P)$. □

We can further reduce the search space for the sets A by bounding $|A|$ from below. For this we observe that we are looking for sets A such that

$$1 > \mathrm{conf}_{\mathbb{K}}(A \to \{\, b \,\}) \geqslant c.$$

Since $A \to \{ b \}$ does not hold in $\mathbb{K}$, there exists at least one object in $A'_{\mathbb{K}}$ that is not contained in $A'_{\mathbb{K}} \cap \{ b \}'_{\mathbb{K}}$, i. e.

$$|A'_{\mathbb{K}}| > |A'_{\mathbb{K}} \cap \{ b \}'_{\mathbb{K}}|.$$

Moreover, the condition $\mathrm{conf}_{\mathbb{K}}(A \to \{ b \}) \geqslant c$ entails

$$|A'_{\mathbb{K}} \cap \{ b \}'_{\mathbb{K}}| \geqslant c \cdot |A'_{\mathbb{K}}|.$$

We can therefore infer that $|A'_{\mathbb{K}}| - 1 \geqslant c \cdot |A'_{\mathbb{K}}|$, or equivalently

$$|A'_{\mathbb{K}}| \geqslant \frac{1}{1-c} \tag{6.11}$$

if $c \neq 1$. Then all sets A which are relevant for the computation of $\mathcal{L}_i^1(P)$ or $\mathcal{L}_i^{1,\mathrm{conf}}(P)$ satisfy this cardinality constraint. Note that for enumerating the sets A that satisfy Equation (6.11), the Next Closure algorithm can be modified accordingly, i. e. the algorithm can be modified in such a way that it enumerates only intents contained in $P''_{\mathbb{K} \div \mathbb{L}_i}$ which satisfy this cardinality constraint. See [48, Theorem 51] for more details on this.

6.2.2. A Non-Approximative Exploration by Confidence

Algorithm 7 has two main flaws: in general, it only provides an approximative exploration of $\mathrm{Th}_c(\mathbb{K})$, and it crucially depends on the computation of closures under $\mathrm{Th}_c(\mathbb{K})$. The latter fact may render the algorithm practically useless, because computing closures under $\mathrm{Th}_c(\mathbb{K})$ can be quite costly.

To achieve an algorithm that indeed implements exploration by confidence, and at the same time avoids the computation of closures under $\mathrm{Th}_c(\mathbb{K})$, we shall make use of Algorithm 6, our algorithm for exploring sets of implications that allows for some freedom in the way implications are asked to the expert. To apply this algorithm to our specific setting, the main problem we have to solve is how to decide whether a given set $P \subseteq M$ is closed under $\mathrm{Th}_c(\mathbb{K})$ or not. Furthermore, we need to define the way the sets P_{i+1} and Q are computed, which eventually will constitute the implication $P_{i+1} \to Q$ which is asked to the expert. As it turns out, we can define the set Q in such a way that $P_{i+1} \to Q$ will always have at least confidence c in $\mathbb{K}$.

Let us start with some wishful thinking. To decide whether P is closed under $\mathcal{L}_i = \mathrm{Th}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i)$ it would be ideal if we could just check whether no element in $P''_{\mathbb{L}_i}$ is entailed by P with confidence at least c, i. e.

$$P = \mathcal{L}_i(P) \iff \forall m \in P''_{\mathbb{L}_i} \backslash P \colon \mathrm{conf}_{\mathbb{K}}(P \to \{ m \}) < c. \tag{6.12}$$

The main benefit would be that we would not need to consider all subsets of P, and thus could avoid this expensive search.

Regrettably, Equation (6.12) is not valid in general, but only the direction from left to right holds. However, we can identify a special case in which this equivalence holds.

6.2.5 Proposition *Let $\mathbb{K} = (G, M, I)$ be a finite formal context, and let $c \in [0, 1]$. Let $\mathbb{L}_i = (G_i, M, I)$ be another finite formal context such that G_i and G are disjoint, and define $\mathcal{L}_i = \mathrm{Th}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i)$. Let $A \subseteq M$ be such that for every intent $X \subsetneq A$ of $\mathbb{K} \div \mathbb{L}_i$ it is true that*

$$\forall m \in X''_{\mathbb{L}_i} \colon \mathrm{conf}_{\mathbb{K}}(X \to \{ m \}) \geqslant c \implies m \in \mathcal{K}_i(X). \tag{6.13}$$

In addition, let A be $\mathcal{K}_i$-closed. Then it is true that A is $\mathcal{L}_i$-closed if and only if

$$A = A''_{\mathbb{K} \div \mathbb{L}_i} \text{ and } \forall m \in A''_{\mathbb{L}_i} \backslash A \colon \mathrm{conf}_{\mathbb{K}}(A \to \{ m \}) < c. \tag{6.14}$$

Proof Let A be $\mathcal{L}_i$-closed. Since $A \to A''_{\mathbb{K} \div \mathbb{L}_i}$ is valid in both $\mathbb{K}$ and $\mathbb{L}_i$, it is true that $(A \to A''_{\mathbb{K} \div \mathbb{L}_i}) \in \mathcal{L}_i$. Therefore, $A = \mathcal{L}_i(A) \supseteq A''_{\mathbb{K} \div \mathbb{L}_i} \supseteq A$ and thus $A = A''_{\mathbb{K} \div \mathbb{L}_i}$ holds. If $m \in A''_{\mathbb{L}_i}$ is such that $\mathrm{conf}_{\mathbb{K}}(A \to \{ m \}) \geqslant c$, then $(A \to \{ m \}) \in \mathcal{L}_i$ and therefore $m \in \mathcal{L}_i(A) = A$ as required.

Conversely, suppose that A is not closed under $\mathcal{L}_i$ and that $A = A''_{\mathbb{K} \div \mathbb{L}_i}$ is true. Then there exists a set $X \subseteq A$ and an attribute $m \in A''_{\mathbb{L}_i} \backslash A$ such that $(X \to \{ m \}) \in \mathcal{L}_i$. Note that then $m \in X''_{\mathbb{L}_i}$, because $X \to \{ m \}$ holds in $\mathbb{L}_i$. Since $A = A''_{\mathbb{K} \div \mathbb{L}_i}$, $X \subseteq A$ implies $X''_{\mathbb{K} \div \mathbb{L}_i} \subseteq A$. Assume by contradiction that $X''_{\mathbb{K} \div \mathbb{L}_i} \subsetneq A$. Then $\mathrm{conf}_{\mathbb{K}}(X''_{\mathbb{K} \div \mathbb{L}_i} \to \{ m \}) = \mathrm{conf}_{\mathbb{K}}(X \to \{ m \}) \geqslant c$ and Equation (6.13) implies

$$m \in \mathcal{K}_i(X''_{\mathbb{K} \div \mathbb{L}_i}) \subseteq \mathcal{K}_i(A) = A,$$

a contradiction. Therefore, $X''_{\mathbb{K} \div \mathbb{L}_i} = A$ and thus $m \in A''_{\mathbb{L}_i} \backslash A$ satisfies $\mathrm{conf}_{\mathbb{K}}(A \to \{ m \}) \geqslant c$ as required. □

Using the same notation as in Algorithm 6, the idea is now to instantiate this algorithm such that Equation (6.13) is satisfied whenever we have to test a $\mathcal{K}_i$-closed set $A \subseteq M$ for being closed under $\mathcal{L}_i$. To achieve this, we shall ask additional questions: in addition to asking the expert implications $P_{i+1} \to Q$ where P_{i+1} is $\mathcal{K}_i$-closed, we shall also ask questions $X \to \{ m \}$, where X is an intent of $\mathbb{K} \div \mathbb{L}_i$, $\mathbb{L}_i$ is the formal context constituted of the counterexamples given so far, and $m \in X''_{\mathbb{L}_i} \backslash \mathcal{K}_i(X)$ satisfies $\mathrm{conf}_{\mathbb{K}}(X \to \{ m \}) \geqslant c$.

This idea is realized in Algorithm 8. Instead of computing the set P_{i+1} directly, we compute there two candidates P^1_{i+1} and P^2_{i+1}. Here, P^2_{i+1} is the "usual" premise we compute for exploration. On the other hand, the idea of considering the sets P^1_{i+1} comes from Equation (6.13) in Proposition 6.2.5: if we are in iteration i, then all intents $X \subsetneq P_i$ of $\mathbb{K} \div \mathbb{L}_i$ had been considered as sets $X = P^1_j$ for some $j < i$, since $X \subsetneq P_i$ implies $X \precneqq P_i$. Then for all $m \in X''_{\mathbb{L}_i} \backslash \mathcal{K}_i(X)$ satisfying $\mathrm{conf}_{\mathbb{K}}(X \to \{ m \}) \geqslant c$, the implication $X \to \{ m \}$ has been asked to the expert. If confirmed, the implication is contained in $\mathcal{K}_i$; if rejected, the context $\mathbb{L}_i$ would contain a counterexample, and thus $m \notin (P_i)''_{\mathbb{L}_i}$. In this way, the condition in Equation (6.13) is ensured.

That this rather informal argumentation indeed holds for Algorithm 8 is shown in the following proposition.

Algorithm 8 (Exploration by Confidence)

define `exploration-by-confidence*`($\mathbb{K} = (G, M, I)$, $\leqslant_M$, p, $c \in [0,1]$, $\mathcal{K} \subseteq \mathrm{Th}(p) \cap \mathrm{Th}_c(\mathbb{K})$)
  $i := 0$, $\mathbb{L}_i = (\varnothing, M, \varnothing)$, $\mathcal{K}_i := \mathcal{K}$, $P_i := \varnothing$

  forever do
    P^1_{i+1} := lectically smallest intent P of $\mathbb{K} \div \mathbb{L}_i$ such that
      $-$ $P_i \preceq P$, and
      $-$ there exists $m \in P''_{\mathbb{L}_i} \backslash \mathcal{K}_i(P)$ such that $\mathrm{conf}_{\mathbb{K}}(P \to \{ m \}) \geqslant c$
      or **nil** if such a set does not exist
    $Q^1_{i+1} := P^1_{i+1} \cup \{ m \}$;; *m from above*

    $P^2_{i+1} :=$ `next-closed-non-closed`$(M, \leqslant_M, P_i, \mathcal{K}_i, \mathbb{K} \div \mathbb{L}_i)$
    $Q^2_{i+1} := (P^2_{i+1})''_{\mathbb{K} \div \mathbb{L}_i}$

    $P_{i+1} := \min_{\preceq}(P^1_{i+1}, P^2_{i+1})$;; **nil** *maximal with respect to* $\preceq$
    if $P_{i+1} =$ **nil exit**

    if $P_{i+1} = P^1_{i+1}$ **then**
      $Q_{i+1} := Q^1_{i+1}$
    else
      $Q_{i+1} := Q^2_{i+1}$
    end

    if $p(P_{i+1} \to Q_{i+1}) = \top$ **then**
      $\mathcal{K}_{i+1} := \mathcal{K}_i \cup \{ P_{i+1} \to Q_{i+1} \}$
      $\mathbb{L}_{i+1} := \mathbb{L}_i$
    else
      $\mathcal{K}_{i+1} := \mathcal{K}_i$
      $\mathbb{L}_{i+1} := \mathbb{L}_i + p(P_{i+1} \to Q_{i+1})$;; *add counterexample to* $\mathbb{L}_i$
    end

    $i := i + 1$
  end

  return $\mathcal{K}_i$
end

6.2.6 Proposition *Suppose that we are in iteration i of Algorithm 8. Then for all intents $X \lneq P_i$ of $\mathbb{K} \div \mathbb{L}_i$ it is true that*

$$\forall m \in X''_{\mathbb{L}_i} \colon \operatorname{conf}_{\mathbb{K}}(X \to \{ m \}) \geqslant c \implies m \in \mathcal{K}_i(X). \tag{6.15}$$

Proof We show the claim by induction over i. For the base case $i = 0$ the claim is vacuously true since $P_0 = \varnothing$. For the step case assume that Equation (6.15) holds for iteration i, and assume further that iteration $i + 1$ exists. Then to show the claim for iteration $i + 1$ let $X \lneq P_{i+1}$ be an intent of $\mathbb{K} \div \mathbb{L}_{i+1}$ and $m \in X''_{\mathbb{L}_i}$ such that $\operatorname{conf}_{\mathbb{K}}(X \to \{ m \}) \geqslant c$. We need to show that $m \in \mathcal{K}_i(X)$ is true. For this, we distinguish two cases.

Case $X \lneq P_i$: If $\mathbb{L}_i = \mathbb{L}_{i+1}$, then X being an intent of $\mathbb{K} \div \mathbb{L}_{i+1}$ also trivially means that X is an intent of $\mathbb{K} \div \mathbb{L}_i$. By induction hypothesis, we obtain $m \in \mathcal{K}_i(X) \subseteq \mathcal{K}_{i+1}(X)$ as required.

If $\mathbb{L}_i \neq \mathbb{L}_{i+1}$, then a counterexample $C \subseteq M$ has been added to $\mathbb{L}_i$ to obtain $\mathbb{L}_{i+1}$, i. e.

$$\mathbb{L}_{i+1} = \mathbb{L}_i + C.$$

The set C is a counterexample to the implication $P_{i+1} \to Q_{i+1}$, and because of this, $P_{i+1} \subseteq C$ is true. Thus, $X \lneq P_{i+1} \preceq C$. Therefore, $C \nsubseteq X$, and it follows that $X''_{\mathbb{L}_{i+1}} = X''_{\mathbb{L}_i}$. Since X is an intent of $\mathbb{K} \div \mathbb{L}_{i+1}$, this implies that X is also an intent of $\mathbb{K} \div \mathbb{L}_i$. Again by induction hypothesis we obtain that $m \in \mathcal{K}_i(X) = \mathcal{K}_{i+1}(X)$.

Case $P_i \preceq X \lneq P_{i+1}$ (note that this case may not occur if $P_i = P_{i+1}$): As argued before, X being an intent of $\mathbb{K} \div \mathbb{L}_{i+1}$ implies that X is also an intent of $\mathbb{K} \div \mathbb{L}_i$. Since $X \lneq P_{i+1} \preceq P^1_{i+1}$, and P^1_{i+1} is the lectically smallest intent of $\mathbb{K} \div \mathbb{L}_i$ such that there exists an element $n \in (P^1_{i+1})''_{\mathbb{L}_i} \setminus \mathcal{K}_i(P^1_{i+1})$ satisfying $\operatorname{conf}_{\mathbb{K}}(P^1_{i+1} \to \{ n \}) \geqslant c$, it follows that $m \in \mathcal{K}_i(X)$, as required. □

To show that Algorithm 8 indeed yields an algorithm that implements exploration by confidence we shall show that it has the form of Algorithm 6. For this we need to show that in every iteration i, the lectically smallest $\mathcal{K}_i$-closed, not $\mathcal{L}_i$-closed set P with $P_i \preceq P$ satisfies

$$P_i \preceq P_{i+1} \preceq P. \tag{6.16}$$

Additionally, we need to show that P_{i+1} is not $\mathcal{L}_i$-closed, and Q_{i+1} satisfies

$$P_{i+1} \subsetneq Q \subseteq \mathcal{L}_i(P_{i+1}), Q \nsubseteq \mathcal{K}_i(P_{i+1}). \tag{6.17}$$

This is enough, as the rest of Algorithm 8 has the same structure as Algorithm 6.

We first show that Equation (6.16) is true. The fact that $P_i \preceq P_{i+1}$ is clear, and $P_{i+1} \preceq P$ is shown in the following proposition.

6.2.7 Proposition *Suppose that we are in iteration i of Algorithm 8. Let $P \subseteq M$ be the lectically smallest $\mathcal{K}_i$-closed set lectically greater or equal to P_i which is not $\mathcal{L}_i$-closed. Then $P_{i+1} \preceq P$.*

Proof We first observe that P^2_{i+1} is $\mathcal{K}_i$-closed by definition, but not $\mathcal{L}_i$-closed, since

$$(P^2_{i+1})''_{\mathbb{K}\div\mathbb{L}_i} \neq P^2_{i+1}$$

because of $\mathrm{Th}(\mathbb{K}\div\mathbb{L}_i) \subseteq \mathcal{L}_i$. This implies $P \preceq P^2_{i+1}$, and the claim holds if $P = P^2_{i+1}$.

It remains to consider the case $P \precneqq P^2_{i+1}$. In this case, by construction of P^2_{i+1} and since P is $\mathcal{K}_i$-closed, the set P must be an intent of $\mathbb{K}\div\mathbb{L}_i$. Since P is not $\mathcal{L}_i$-closed, there must exist a set $\overline{P} \subseteq P$ and an element $m \in P''_{\mathbb{L}_i}\backslash P$ such that

$$\mathrm{conf}_{\mathbb{K}}(\overline{P} \to \{m\}) \geqslant c,$$

and $\overline{P} \to \{m\}$ is valid in $\mathbb{L}_i$. In particular, $m \in \overline{P}''_{\mathbb{L}_i}\backslash\mathcal{K}_i(\overline{P})$.

Since P is an intent of $\mathbb{K}\div\mathbb{L}_i$ we can assume that $\overline{P}$ is also an intent of $\mathbb{K}\div\mathbb{L}_i$. If $\overline{P} \precneqq P_i$, then by Proposition 6.2.6 it would be true that $m \in \mathcal{K}_i(\overline{P}) \subseteq \mathcal{K}_i(P) = P$, a contradiction. Therefore, $P_i \preceq \overline{P}$. By definition of P^1_{i+1} it holds that $P^1_{i+1} \preceq \overline{P}$, and thus $P^1_{i+1} \preceq \overline{P} \preceq P$, because $\overline{P} \subseteq P$. Thus, $P_{i+1} \preceq P$, as required. ▫

Furthermore, it is easy to see that P_{i+1} is not $\mathcal{L}_i$-closed: if $P_{i+1} = P^1_{i+1}$, then the element m found during this computation satisfies $m \in \mathcal{L}_i(P_{i+1})\backslash P_{i+1}$. If $P_{i+1} = P^2_{i+1}$, then $P^2_{i+1} \neq (P^2_{i+1})''_{\mathbb{K}\div\mathbb{L}_i}$ shows that P_{i+1} is not $\mathcal{L}_i$-closed.

It remains to show that Q_{i+1} satisfies Equation (6.17), i. e.

$$P_{i+1} \subsetneq Q_{i+1} \subseteq \mathcal{L}_i(P_{i+1}).$$

To see this, first suppose that $P_{i+1} = P^1_{i+1}$. Then Q contains an attribute m that satisfies $m \notin \mathcal{K}_i(P_{i+1}) \supseteq P_{i+1}$. In this case, $\mathrm{conf}_{\mathbb{K}}(P^1_{i+1} \to \{m\}) \geqslant c$ and $m \in P''_{\mathbb{L}_i}$, thus $(P_{i+1} \to \{m\}) \in \mathcal{L}_i$ and $Q_{i+1} \subseteq \mathcal{L}_i(P_{i+1})$. On the other hand, if $P_{i+1} = P^2_{i+1}$, then $P_{i+1} \neq (P_{i+1})''_{\mathbb{K}\div\mathbb{L}_i} = Q_{i+1}$. Since $\mathrm{Th}(\mathbb{K}\div\mathbb{L}_i) \subseteq \mathcal{L}_i$, $Q_{i+1} \subseteq \mathcal{L}_i(P_{i+1})$ is true. Finally, $Q_{i+1} \nsubseteq \mathcal{K}_i(P^2_{i+1})$ is clear since $\mathcal{K}_i(P^2_{i+1}) = P^2_{i+1}$ and $Q \nsupseteq P^2_{i+1}$.

We have thus shown that Algorithm 8 has the form of Algorithm 6. The following result is then an immediate consequence of Theorem 6.1.17 and Theorem 6.1.19. Note that our algorithm only asks implications which are either valid in $\mathbb{K}$, or at least have confidence at least c in $\mathbb{K}$. Thus, $\mathcal{K}_i\backslash\mathcal{K} \subseteq \mathrm{Th}_c(\mathbb{K})$ holds in every iteration i. Therefore, in contrast to Algorithm 7, Algorithm 8 always computes a base of $\mathrm{Th}(p) \cap \mathrm{Th}_c(\mathbb{K})$.

6.2.8 Corollary *Let $\mathbb{K} = (G, M, I)$ be a finite formal context, $\leqslant_M$ a linear order on M, p a domain expert on M, $c \in [0,1]$ and $\mathcal{K} \subseteq \mathrm{Th}(p) \cap \mathrm{Th}_c(\mathbb{K})$. Then Algorithm 8 applied to this input terminates after finitely many steps. If n is the number of iterations of this run, then $\mathcal{K}_n\backslash\mathcal{K}$ is a confident base of $\mathrm{Th}(p) \cap \mathrm{Th}_c(\mathbb{K})$ with background knowledge $\mathcal{K}$, i. e. $\mathcal{K}_n\backslash\mathcal{K} \subseteq \mathrm{Th}_c(\mathbb{K})$ and*

$$\mathrm{Cn}(\mathcal{K}_n) = \mathrm{Cn}(\mathrm{Th}(p) \cap \mathrm{Th}_c(\mathbb{K})).$$

If $\mathbb{L}$ is the formal context that consists of all counterexamples provided by p during the exploration, then for each $(A \to B) \in \mathrm{Th}_c(\mathbb{K})$ it is true that either $(A \to B) \in \mathrm{Cn}(\mathcal{K}_n)$ or $(A \to B) \notin \mathrm{Th}(\mathbb{L})$.

Chapter 7

Model Exploration by Confidence with Completely Specified Counterexamples

We have seen how we can extend attribute exploration to explore implications which enjoy a high confidence in some given formal context. In this chapter we want to extend this generalization of attribute exploration even further to be able to explore GCIs with high confidence in finite interpretations. The basis for this extension will be the *model exploration* algorithm from [41], an extension of attribute exploration to explore valid GCIs of finite interpretations.

Model exploration, similarly to attribute exploration, assumes that a certain domain of interest is representable by a finite interpretation $\mathcal{I}_{\mathsf{back}}$, which we shall call the *background interpretation* of the exploration process. However, we assume that this interpretation is not directly accessible, but instead an expert is given that allows us to decide whether certain GCIs are valid in $\mathcal{I}_{\mathsf{back}}$. In addition, if a given GCI $C \sqsubseteq D$ is not valid in $\mathcal{I}_{\mathsf{back}}$, then the expert can provide counterexamples for $C \sqsubseteq D$ in a suitable way.

The principal way how model exploration works is again very akin to attribute exploration. Given a finite *connected subinterpretation* $\mathcal{I}$ of $\mathcal{I}_{\mathsf{back}}$ and a set $\mathcal{B}$ of valid GCIs of $\mathcal{I}_{\mathsf{back}}$, the algorithm successively generates valid GCIs $C \sqsubseteq D$ of $\mathcal{I}$, which do not follow from $\mathcal{B}$, and presents them to some expert. If the expert confirms $C \sqsubseteq D$, then it is added to $\mathcal{B}$. If the expert rejects $C \sqsubseteq D$, then she provides a counterexample in the form of a connected subinterpretation of $\mathcal{I}_{\mathsf{back}}$, which is added to $\mathcal{I}$. Since $\mathcal{I}$ and $\mathcal{B}$ play the same role as the working context and the set of known implications during attribute exploration, we shall refer to them as the *working interpretation* and the *set of known GCIs*, respectively. If no more GCIs can be generated to be asked to the expert, the algorithm stops. It can be shown that at this point, the set $\mathcal{B}$ is a finite base of $\mathcal{I}_{\mathsf{back}}$.

The foregone description already suggests that there are some difficulties in transferring attribute exploration to the setting of GCIs and finite interpretations. The most apparent is that the set of GCIs we potentially have to cover is infinite, as the set of valid GCIs of $\mathcal{I}_{\mathsf{back}}$ is infinite.

Another problem is that validity of GCIs in interpretations deploys a *closed-world* semantics: if an element $x \in \Delta^{\mathcal{I}}$ of a (finite) interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ does not have an r-successor in $\mathcal{I}$ for some $r \in N_R$, then it is assumed that x does not have r-successors $\mathcal{I}_{\mathsf{back}}$. Therefore, if we add an element x of $\mathcal{I}_{\mathsf{back}}$ as a counterexample for a GCI to our working interpretation $\mathcal{I}$, then we also have to include *all* its role successors (and their role successors, and so on) in $\mathcal{I}_{\mathsf{back}}$, nevertheless they may not be necessary for the counterexample; otherwise, elements in $\mathcal{I}$ may serve as counterexamples to GCIs which are valid in $\mathcal{I}_{\mathsf{back}}$, because missing information is considered as false information. The approach followed by Baader and Distel to account for this problem is to let the expert provide *connected subinterpretations* of $\mathcal{I}_{\mathsf{back}}$ as counterexamples for GCIs.

We shall discuss the details of model exploration in Section 7.1. Based on this discussion, we shall develop a model exploration algorithm that also includes GCIs with high confidence among those proposed to the expert. This algorithm, which we shall call *model exploration by confidence*, will be introduced in Section 7.2, and its construction will mimic the argumentation used by Baader and Distel for their model exploration algorithm.

The results presented in this section have been published previously in [31].

7.1. Model Exploration with Valid GCIs

In this section we shall review the argumentation used to develop model exploration, as given in [41, Chapter 6]. In the next section, we shall use this argumentation presented here and generalize it to the setting of GCIs with high confidence in finite interpretations.

Model exploration is based on the result that bases of finite interpretations $\mathcal{I}$ can be obtained from bases of their corresponding induced formal context $\mathbb{K}_{\mathcal{I}}$ (Theorem 4.3.7). Since attribute exploration arises from the computation of the canonical base by adding suitable expert interaction (see Section 2.5), one could think of obtaining an algorithm for model exploration by adding suitable expert interaction during the computation of bases of $\mathbb{K}_{\mathcal{I}}$. It shall turn out that this is indeed correct.

However, there is a technical problem which does not arise in attribute exploration: when we add counterexamples to our current working interpretation $\mathcal{I}$ during model exploration, then the attribute set $M_{\mathcal{I}}$ of the corresponding induced context $\mathbb{K}_{\mathcal{I}}$ may change, since it depends on the elements of $\mathcal{I}$. Recall that $M_{\mathcal{I}}$ was defined as

$$M_{\mathcal{I}} = N_C \cup \{ \bot \} \cup \{ \exists r.X^{\mathcal{I}} \mid r \in N_R, X \subseteq \Delta^{\mathcal{I}}, X \neq \varnothing \}.$$

Thus, to allow to use attribute exploration as a basis for model exploration, we need to fix the attribute set, and the best way for this would be to use $M_{\mathcal{I}_{\mathsf{back}}}$. However, since we cannot access the background interpretation $\mathcal{I}_{\mathsf{back}}$ directly, we cannot compute this set completely. On the other hand, it can be shown that we can compute the set $M_{\mathcal{I}_{\mathsf{back}}}$ *incrementally*, using the fact that the expert confirms certain types of GCIs, and then use the parts of $M_{\mathcal{I}_{\mathsf{back}}}$ we already know for the exploration process.

To explain how this can be done, we shall first discuss in Section 7.1.1 how we can compute bases of formal contexts where the set of attributes is allowed to grow during the computation. Thereafter, we shall see in Section 7.1.2 how we can transfer this algorithm to the setting of computing finite bases of finite interpretations $\mathcal{I}$, thus allowing the set $M_{\mathcal{I}}$ to be computed successively during the computation. Finally, we shall see in Section 7.1.3 how we can add expert interaction to avoid direct access to the underlying interpretation, thus obtaining the model exploration algorithm.

7.1.1. Growing Sets of Attributes

We want to find an algorithm that allows us to compute bases of formal contexts where the attribute set is allowed to grow during the computation. We can think of this situation as follows: we want to compute a base of a formal context, which we cannot access completely, in the sense that some of the attributes in this formal context are "hidden". However, during the computation of the base, hidden attributes are uncovered incrementally. The goal is then to find an algorithm which allows us to compute bases in such a setting.

Obtaining such an algorithm is actually not that difficult. For this let us consider how Algorithm 2 computes the canonical base. There, we use the Next-Closure algorithm to enumerate the premises of the canonical base of a given formal context $\mathbb{K} = (G, M, I)$, using some linear order $\leqslant_M$ on M. If $M = \{ m_1, \ldots, m_n \}$ and

$$m_n \leqslant_M m_{n-1} \leqslant_M \cdots \leqslant_M m_1,$$

then the Next Closure algorithm firstly enumerates all premises which are subsets of $\varnothing$, then those which are subsets of $\{ m_1 \}$, then those of $\{ m_1, m_2 \}$, and so on. In particular, it will not consider an element $m_k \in M$ before it has enumerated all premises which are subsets of $\{ m_1, \ldots, m_{k-1} \}$.

We can exploit this idea for our purpose of computing bases with growing sets of attributes: if the attribute set in iteration i is M_i, ordered by $\leqslant_{M_i}$, and we are about to add some new attributes $m_1, \ldots, m_n$ to M_i to obtain

$$M_{i+1} := M_i \cup \{ m_1, \ldots, m_n \},$$

then we define the linear order $\leqslant_{M_{i+1}}$ on M_{i+1} by ordering the elements in $M_i \subseteq M_{i+1}$ as before, i. e.

$$\leqslant_{M_i} = \leqslant_{M_{i+1}} \cap\, M_i \times M_i, \tag{7.1}$$

and requiring in addition that

$$m_j \leqslant_{M_{i+1}} x \tag{7.2}$$

is true for all $j \in \{ 1, \ldots, n \}$ and $x \in M_i$. In other words, we just put the new elements *before* the old elements. In that way, the Next-Closure behaves as if the elements would have been there from the start, and computes the base as desired.

Algorithm 9 (Algorithm 8 from [41]) Computing a Base of a Formal Context with Growing Sets of Attributes and Background Knowledge

define `base/growing-set-of-attributes`($\mathbb{K} = (G, M, I), \leqslant_M, \mathcal{S} \subseteq \mathrm{Th}(\mathbb{K})$)
  $i := 0$
  $P_i := \emptyset$
  $\mathcal{K}_i := \emptyset$
  $\mathbb{K}_i := \mathbb{K}$
  $M_i := M$
  $\mathcal{S}_i := \mathcal{S}$
  $\leqslant_{M_i} := \leqslant_M$

  forever do
    read $\mathbb{K}_{i+1} = (G, M_{i+1}, I_{i+1})$ such that $M_i \subseteq M_{i+1}$ and $I_i = I_{i+1} \cap M_i \times M_i$
    read $\mathcal{S}_{i+1}$ such that $\mathcal{S}_i \subseteq \mathcal{S}_{i+1} \subseteq \mathrm{Th}(\mathbb{K}_{i+1})$
    choose $\leqslant_{M_{i+1}}$ such that (7.1) and (7.2) hold.

    $\mathcal{K}_{i+1} := \{ P_r \to (P_r)''_{\mathbb{K}_{i+1}} \mid P_r \neq (P_r)''_{\mathbb{K}_{i+1}}, r \in \{0, \ldots, i\} \}$

    $P_{i+1} :=$ `next-closure`$(M_{i+1}, \leqslant_{M_{i+1}}, P_i, \mathcal{K}_{i+1} \cup \mathcal{S}_{i+1})$
    if $P_{i+1} =$ **nil exit**

    $i := i + 1$
  end

  return $\mathcal{K}_i$
end

An implementation of this idea is shown in Algorithm 9. There we start with some initial formal context $\mathbb{K}_0 = \mathbb{K} = (G, M, I)$ and some background knowledge $\mathcal{S}_0 = \mathcal{S} \subseteq \mathrm{Th}(\mathbb{K})$. Then, in every iteration we allow to extend the current context $\mathbb{K}_i = (G, M_i, I_i)$ by providing a new set $M_{i+1} \supseteq M_i$ of attributes and a new incidence relation $I_{i+1} \subseteq M_{i+1} \times M_{i+1}$ which satisfies

$$I_i = I_{i+1} \cap M_i \times M_i.$$

This corresponds to our perception that at the beginning of the run of the algorithm, some of the attributes are hidden, and are uncovered during the run.

For this algorithm to make sense, we of course require that at a certain point everything from the formal context has been uncovered, i. e. that for some $\ell \in \mathbb{N}_{\geqslant 0}$ it is true that $M_\ell = M_i$ for all $i \geqslant \ell$. From this point on, Algorithm 9 behaves like Algorithm 2 for computing the canonical base of a given formal context.

7.1.1 Theorem (Theorems 6.2 and 6.3 from [41]) *Let $\mathbb{K} = (G, M, I)$ be a finite formal context, $\leqslant_M$ a linear order on M, and $\mathcal{S} \subseteq \mathrm{Th}(\mathbb{K})$. Then in a run of Algorithm 9, let $\ell \in \mathbb{N}_{\geqslant 0}$ be such that $M_\ell = M_i$ for all $i \geqslant \ell$. Then this run terminates. If n is the last iteration of this run, then $\mathcal{K}_n$ is a base of $\mathbb{K}_n$ with background knowledge $\mathcal{S}_n$.*

A difference to the classical computation of the canonical base as shown in Algorithm 2 is that in the latter we only consider sets P as premises for implications which are closed under the currently known implications, but are not intents of the given formal context. In contrast to this, Algorithm 9 considers all sets P_i which are closed under the currently known implications, no matter whether they are intents of $\mathbb{K}_i$. The reason for this is that even if P_i is an intent of $\mathbb{K}_i$, it could very well be that P_i is not an intent of $\mathbb{K}_n$ (where n is the number of iterations of the algorithm) because of attributes which have been introduced in $\mathbb{K}_n$, but were not present in $\mathbb{K}_i$. Since we cannot know whether P_i will be an intent of $\mathbb{K}_n$ or not, when we compute it, we have to consider it as well. Otherwise, we cannot guarantee that $\mathcal{K}_n$ will be a base of $\mathbb{K}_n$.

Unfortunately, the fact that we have to keep all those sets P_i may lead to $\mathcal{K}_n$ not being irredundant anymore. This has been illustrated in [41] by the following example.

7.1.2 Example (Example 6.1 from [41]) We consider the following run of Algorithm 9 with input $\mathbb{K} = \mathbb{K}_0$ as shown in Figure 7.1, and $\mathcal{S} = \mathcal{S}_0 = \varnothing = \mathcal{S}_1 = \ldots = \mathcal{S}_6$:

k	$M_{k+1} \setminus M_k$	$\mathcal{L}_k$	P_k
0	$\varnothing$	$\varnothing$	$\varnothing$
1	$\varnothing$	$\varnothing$	$\{\mathsf{A}\}$
2	$\{\mathsf{B}\}$	$\varnothing$	$\{\mathsf{B}\}$
3	$\varnothing$	$\varnothing$	$\{\mathsf{A},\mathsf{B}\}$
4	$\{\mathsf{C}\}$	$\{\{\mathsf{A}\} \to \{\mathsf{A},\mathsf{C}\}, \{\mathsf{A},\mathsf{B}\} \to \{\mathsf{A},\mathsf{B},\mathsf{C}\}\}$	$\{\mathsf{C}\}$
5	$\varnothing$	$\{\{\mathsf{A}\} \to \{\mathsf{A},\mathsf{C}\}, \{\mathsf{A},\mathsf{B}\} \to \{\mathsf{A},\mathsf{B},\mathsf{C}\}, \{\mathsf{C}\} \to \{\mathsf{A},\mathsf{C}\}\}$	$\{\mathsf{A},\mathsf{B},\mathsf{C}\}$
6	$\varnothing$	$\{\{\mathsf{A}\} \to \{\mathsf{A},\mathsf{C}\}, \{\mathsf{A},\mathsf{B}\} \to \{\mathsf{A},\mathsf{B},\mathsf{C}\}, \{\mathsf{C}\} \to \{\mathsf{A},\mathsf{C}\}\}$	**nil**

$\mathbb{K}_0 = \mathbb{K}_1 =$

	A
1	×
2	

$\mathbb{K}_2 = \mathbb{K}_3 =$

	A	B
1	×	
2		×

$\mathbb{K}_3 = \mathbb{K}_4 = \mathbb{K}_5 =$

	A	B	C
1	×		×
2		×	

Figure 7.1.: Formal Contexts for Example 7.1.2

In iterations 2 and 4, the new attributes B and C are added, as shown in Figure 7.1. The algorithm terminates in iteration 6 with output $\mathcal{L}_6$, which is clearly non-redundant: the implication $\{ \mathsf{A}, \mathsf{B} \} \to \{ \mathsf{A}, \mathsf{B}, \mathsf{C} \}$ is entailed by $\{ \mathsf{A} \} \to \{ \mathsf{A}, \mathsf{C} \}$. ♢

7.1.2. Computing Bases of Given Finite Interpretations

We now want to use Algorithm 9 to devise an algorithm that allows us to compute bases of finite interpretations $\mathcal{I}$ without computing $M_\mathcal{I}$ first. Instead, we want that the elements of the set $M_\mathcal{I}$ are computed successively during the run of the algorithm. In this way, we can immediately start with computing valid GCIs of $\mathcal{I}$, and do not have to wait for $M_\mathcal{I}$ to be computed completely.

The successive computation of the elements of $M_\mathcal{I}$ is achieved in Algorithm 9 by defining the sets M_i as follows. For $i = 0$, we define

$$M_0 = N_C \cup \{ \bot \}.$$

Then, during the run of the algorithm, we add elements of the form $\exists r.X^\mathcal{I}$ for $r \in N_R$ and $X \subseteq \Delta^\mathcal{I}$, $X \neq \emptyset$. More precisely, whenever we compute a set P_i in Algorithm 9, we define

$$M_{i+1} := M_i \cup \{ \exists r.(\textstyle\bigsqcap P_i)^{\mathcal{I}\mathcal{I}} \mid r \in N_R \},$$

where the union is only up to equivalence, i. e. if for some $C := \exists r.(\bigsqcap P_i)^{\mathcal{I}\mathcal{I}}$ there already exists a $D \in M_i$ such that $C \equiv D$, then we do not add C in the definition of M_{i+1}.

We also need to specify how we define the formal contexts $\mathbb{K}_i$ and the sets $\mathcal{S}_i$ of background knowledge. We set $\mathbb{K}_i$ to be the induced formal context of M_i and $\mathcal{I}$, and we define

$$\mathcal{S}_{i+1} = \{ \{ A \} \to \{ B \} \mid A, B \in M_i, A \sqsubseteq B \}.$$

The resulting algorithm is shown in Algorithm 10.

Note that Algorithm 10 has the form of Algorithm 9, and thus we can argue that a run of Algorithm 10 terminates with finite sets N_C, N_R and a finite interpretation $\mathcal{I}$ as input. In this case, all concept descriptions which are added to the set of attributes during the run of

Algorithm 10 (Algorithm 9 from [41]) Computing a Base of a Given Interpretation with Incremental Computation of $M_{\mathcal{I}}$

0 **define** `base-of-interpretation`($\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ over N_C and N_R)
1 $i := 0$
2 $P_i := \varnothing$
3 $M_i := N_C \cup \{ \bot \}$
4 $\mathcal{K}_i := \varnothing$
5 $\mathcal{S}_i := \{ \{ \bot \} \to \{ A \} \mid A \in N_C \}$
6 **choose** $\leqslant_{M_i}$ as a linear order on M_i
7
8 **forever do**
9 $M_{i+1} := M_i \cup \{ \exists r.(\bigsqcap P_i)^{\mathcal{I}\mathcal{I}} \mid r \in N_R \}$
10 $\mathbb{K}_{i+1} :=$ `induced-context`($\mathcal{I}$, M_{i+1})
11 $\mathcal{K}_{i+1} := \{ P_r \to (P_r)''_{\mathbb{K}_{i+1}} \mid P_r \neq (P_r)''_{\mathbb{K}_{i+1}}, r \in \{ 0, \ldots, i \} \}$
12 $\mathcal{S}_{i+1} := \{ \{ A \} \to \{ B \} \mid A, B \in M_{i+1}, A \sqsubseteq B \}$
13 **choose** $\leqslant_{M_{i+1}}$ such that (7.1) and (7.2) hold.
14
15 $P_{i+1} :=$ `next-closure`($M_{i+1}, \leqslant_{M_{i+1}}, P_i, \mathcal{K}_{i+1} \cup \mathcal{S}_{i+1}$)
16 **if** $P_{i+1} =$ **nil exit**
17
18 $i := i + 1$
19 **end**
20
21 **return** $\{ \bigsqcap P \sqsubseteq (\bigsqcap P)^{\mathcal{I}\mathcal{I}} \mid (P \to P''_{\mathbb{K}_{i+1}}) \in \mathcal{K}_{i+1} \}$
22 **end**

the algorithm are, up to equivalence, elements of $M_{\mathcal{I}}$, which is finite. Thus, there exists a number $\ell \in \mathbb{N}_{\geqslant 0}$ such that for all $i \geqslant \ell$ it is true that $M_i = M_\ell$. Then, by Theorem 7.1.1, Algorithm 10 has to terminate.

To see that the definition of M_i will eventually yield all elements of $M_{\mathcal{I}}$, up to equivalence, we first observe that $M_i \subseteq M_{\mathcal{I}}$ is true up to equivalence for all iterations i of Algorithm 10. On the other hand, if $\exists r.X^{\mathcal{I}} \in M_{\mathcal{I}}$, then $X^{\mathcal{I}} \equiv X^{\mathcal{I}\mathcal{I}\mathcal{I}} = (X^{\mathcal{I}})^{\mathcal{I}\mathcal{I}}$, and $X^{\mathcal{I}}$ is expressible in terms of $M_{\mathcal{I}}$ by Lemma 4.3.5. Therefore, there exists $U \subseteq M_{\mathcal{I}}$ such that

$$X^{\mathcal{I}} \equiv \bigsqcap U.$$

If n is the number of iterations of the algorithm, and if $\mathbb{K}_n$ denotes the induced context of M_n and $\mathcal{I}$, then we find

$$(\bigsqcap U''_{\mathbb{K}_n})^{\mathcal{I}} = U'''_{\mathbb{K}_n} = U'_{\mathbb{K}_n} = (\bigsqcap U)^{\mathcal{I}}$$

using Proposition 4.2.9. Then

$$\exists r.X^{\mathcal{I}} \equiv \exists r.(X^{\mathcal{I}})^{\mathcal{I}\mathcal{I}} \equiv \exists r.(\bigsqcap U''_{\mathbb{K}_n})^{\mathcal{I}\mathcal{I}}.$$

Thus, it suffices to consider only intents of the final context $\mathbb{K}_n$. The following result shows that these intents are among the sets P_i.

7.1.3 Lemma (Partly Lemma 6.3 from [41]) *Consider a terminating run of Algorithm 10 with n iterations, and let $Q \subseteq M_n$. Then if $Q = Q''_{\mathbb{K}_n}$, then $Q = P_i$ for some $i \in \{0, \ldots, n\}$.*

Using this lemma we can show that M_n is, up to equivalence, equal to $M_{\mathcal{I}}$. It can then be shown that a base of the induced context of $\mathcal{I}$ and M_n yields a base of $\mathcal{I}$ as well [41, Corollary 5.14]. From this, we immediately obtain the correctness of Algorithm 10.

7.1.4 Theorem (Theorem 6.9 from [41]) *Let $\mathcal{I}$ be a finite interpretation over N_C and N_R. Then the set*

$$\mathcal{K} = \texttt{base-of-interpretation}(\mathcal{I})$$

is a finite base of $\mathcal{I}$.

7.1.3. An Algorithm for Exploring Interpretations

Based on Algorithm 10, we now want to discuss an algorithm for model exploration. For this, recall that during model exploration we suppose that our domain of interest can be represented by a finite interpretation $\mathcal{I}_{\mathsf{back}}$, the background interpretation of the exploration. If we could access $\mathcal{I}_{\mathsf{back}}$ directly, then to explore $\mathcal{I}_{\mathsf{back}}$ would just mean to compute a base of it, which we could achieve by using Algorithm 10. However, as already discussed, we assume that $\mathcal{I}_{\mathsf{back}}$ cannot be accessed directly, but instead is represented by an expert.

To turn Algorithm 10 into an algorithm that allows us to compute a base of $\mathcal{I}_{\mathsf{back}}$ using only the expert as a means to access this interpretation, we want to replace every direct access to $\mathcal{I}_{\mathsf{back}}$ in Algorithm 10 by a suitable expert interaction. For this we observe that there are two places in Algorithm 10 that directly access the given interpretation:

i. when computing concept descriptions of the form $\exists r.(\bigsqcap P_i)^{\mathcal{I}_{\mathsf{back}}}$ (line 9 of Algorithm 10),

ii. when computing $\mathbb{K}_i$ as induced context of M_i and $\mathcal{I}_{\mathsf{back}}$ (line 10 of Algorithm 10).

The computation of $\mathbb{K}_i$ we can fix easily if instead of computing the induced context of M_i and $\mathcal{I}_{\mathsf{back}}$, we just compute the induced context M_i and the current working interpretation of the exploration process. For computing $\exists r.(\bigsqcap P_i)^{\mathcal{I}_{\mathsf{back}}}$, however, we need to use the expert.

For this, we need to consider another issue first, which we have already talked about in the introduction, namely the way the experts specifies counterexamples during the exploration. We had argued that if the expert gives an element $x \in \Delta^{\mathcal{I}_{\mathsf{back}}}$ from the background interpretation as a counterexample, then she also has to include all corresponding concept names and role successors x has in $\mathcal{I}_{\mathsf{back}}$. Otherwise, there is the risk that the provided counterexamples invalidate GCIs which are actually valid in the background interpretation $\mathcal{I}_{\mathsf{back}}$.

To make this more formal, Distel introduced the notion of a *connected subinterpretation*.

7.1.5 Definition (Connected Subinterpretations; Definition 6.1 from [41]) Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation over N_C and N_R. Define

$$\begin{aligned} \mathrm{names}_{\mathcal{I}}(x) &:= \{\, C \in N_C \mid x \in C^{\mathcal{I}} \,\}, \\ \mathrm{succ}_{\mathcal{I}}(x,r) &:= \{\, y \in \Delta^{\mathcal{I}} \mid (x,y) \in r^{\mathcal{I}} \,\}, \end{aligned}$$

for $x \in \Delta^{\mathcal{I}}$ and $r \in N_R$. An interpretation $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ over N_C and N_R is called a *subinterpretation* of $\mathcal{I}$ if and only if

i. $\Delta^{\mathcal{J}} \subseteq \Delta^{\mathcal{I}}$,

ii. $\mathrm{names}_{\mathcal{I}}(x) = \mathrm{names}_{\mathcal{J}}(x)$ for all $x \in \Delta^{\mathcal{J}}$, and

iii. $\mathrm{succ}_{\mathcal{J}}(x,r) \subseteq \mathrm{succ}_{\mathcal{I}}(x,r)$ for all $x \in \Delta^{\mathcal{J}}, r \in N_R$.

$\mathcal{J}$ is called a *connected subinterpretation* of $\mathcal{I}$ if $\mathcal{J}$ is a subinterpretation of $\mathcal{I}$, and in addition it is true that

$$\mathrm{succ}_{\mathcal{J}}(x,r) = \mathrm{succ}_{\mathcal{I}}(x,r)$$

is true for all $x \in \Delta^{\mathcal{J}}, r \in N_R$. In this case we shall say that $\mathcal{I}$ *extends* $\mathcal{J}$. ♢

If we now ensure that during the exploration process the current working interpretation is a connected subinterpretation of the background interpretation $\mathcal{I}_{\mathsf{back}}$, then we can guarantee that counterexamples provided by the expert do not accidentally invalidate valid GCIs. This can be achieved by adding counterexamples only as connected subinterpretations of $\mathcal{I}_{\mathsf{back}}$ to our current working interpretation.

7.1.6 Lemma (Lemma 6.12 from [41]) *Let $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ be an interpretation over N_C and N_R which is a connected subinterpretation of the interpretation $\mathcal{I}$. Then for all $\mathcal{EL}^{\bot}_{gfp}$ concept descriptions C over N_C and N_R it is true that*

$$C^{\mathcal{J}} = C^{\mathcal{I}} \cap \Delta^{\mathcal{J}}.$$

7.1.7 Theorem (Corollary 6.13 from [41]) *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation over N_C and N_R, and let $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ be a connected subinterpretation of $\mathcal{I}$. Let C, D be two $\mathcal{EL}^{\bot}_{gfp}$ concept descriptions over N_C and N_R. Then if $C \sqsubseteq D$ is valid in $\mathcal{I}$, then $C \sqsubseteq D$ is also valid in $\mathcal{J}$.*

Proof Since $C \sqsubseteq D$ holds in $\mathcal{I}$, it is true that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Using Lemma 7.1.6 we thus obtain

$$C^{\mathcal{J}} = C^{\mathcal{I}} \cap \Delta^{\mathcal{J}} \subseteq D^{\mathcal{I}} \cap \Delta^{\mathcal{J}} = D^{\mathcal{J}},$$

i. e. $C \sqsubseteq D$ holds in $\mathcal{J}$, as it was claimed. □

Now that we know how the expert should provide counterexamples to proposed GCIs, let us reconsider the question of how to compute concept descriptions of the form $\exists r.(\bigsqcap P_i)^{\mathcal{I}_{\mathsf{back}}\mathcal{I}_{\mathsf{back}}}$. Recall that since we cannot access $\mathcal{I}_{\mathsf{back}}$, we cannot compute this concept description directly. What we can compute is the concept description $\exists r.(\bigsqcap P_i)^{\mathcal{I}_\ell\mathcal{I}_\ell}$, where $\mathcal{I}_\ell$ is the currently known interpretation. The good thing is that the expert can ensure that $\exists r.(\bigsqcap P_i)^{\mathcal{I}_{\mathsf{back}}\mathcal{I}_{\mathsf{back}}}$ and $\exists r.(\bigsqcap P_i)^{\mathcal{I}_\ell\mathcal{I}_\ell}$ are equivalent.

7.1.8 Lemma (Lemma 6.14 from [41]) *Let $\mathcal{I}$ be a finite interpretation over N_C and N_R, and let $\mathcal{J}$ be a connected subinterpretation of $\mathcal{I}$. Then for all $\mathcal{EL}^{\bot}_{gfp}$ concept descriptions C over N_C and N_R, it is true that if $C \sqsubseteq C^{\mathcal{J}\mathcal{J}}$ is valid in $\mathcal{I}$, then $C^{\mathcal{I}\mathcal{I}} \equiv C^{\mathcal{J}\mathcal{J}}$.*

If we choose $\mathcal{I} = \mathcal{I}_{\mathsf{back}}$, $\mathcal{J} = \mathcal{I}_\ell$ and $C = \bigsqcap P_i$ in the previous lemma we see that if the expert confirms the GCI

$$\bigsqcap P_i \sqsubseteq (\bigsqcap P_i)^{\mathcal{I}_\ell\mathcal{I}_\ell},$$

then $(\bigsqcap P_i)^{\mathcal{I}_\ell\mathcal{I}_\ell} \equiv (\bigsqcap P_i)^{\mathcal{I}_{\mathsf{back}}\mathcal{I}_{\mathsf{back}}}$, just as we need it.

On the other hand, if the expert rejects $\bigsqcap P_i \sqsubseteq (\bigsqcap P_i)^{\mathcal{I}_\ell\mathcal{I}_\ell}$, then she adds counterexamples to the current working interpretation $\mathcal{I}_\ell$ to yield a new working interpretation $\mathcal{I}_{\ell+1}$. If $\bigsqcap P_i \not\sqsubseteq (\bigsqcap P_i)^{\mathcal{I}_{\ell+1}\mathcal{I}_{\ell+1}}$, then the GCI

$$\bigsqcap P_i \sqsubseteq (\bigsqcap P_i)^{\mathcal{I}_{\ell+1}\mathcal{I}_{\ell+1}}$$

Algorithm 11 (Algorithm 11 from [41]) A Model Exploration Algorithm

0 **define** `model-exploration`($\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ over N_C and N_R)
1 $i := 0$
2 $P_i := \varnothing$
3 $M_i := N_C \cup \{ \bot \}$
4 $\mathcal{K}_i := \varnothing$
5 $\mathcal{S}_i := \{ \{ \bot \} \to \{ A \} \mid A \in N_C \}$
6 **choose** $\leqslant_{M_i}$ as a linear order on M_i
7 $\ell := 0$
8 $\mathcal{I}_\ell := \mathcal{I}$
9
10 **forever do**
11 *;; expert interaction*
12 **while** expert refutes $\bigcap P_i \sqsubseteq (\bigcap P_i)^{\mathcal{I}_\ell \mathcal{I}_\ell}$ **do**
13 $\mathcal{I}_{\ell+1}$:= new interpretation such that
14 $-$ $\mathcal{I}_{\ell+1}$ extends $\mathcal{I}_\ell$
15 $-$ $\mathcal{I}_{\ell+1}$ contains counterexamples for $\bigcap P_i \sqsubseteq (\bigcap P_i)^{\mathcal{I}_\ell \mathcal{I}_\ell}$
16 $\ell := \ell + 1$
17 **end**
18
19 *;; add new attributes (up to equivalence)*
20 $M_{i+1} := M_i \cup \{ \exists r.(\bigcap P_i)^{\mathcal{I}_\ell \mathcal{I}_\ell} \mid r \in N_R \}$
21
22 *;; update* $\mathbb{K}_{i+1}, \mathcal{S}_{i+1}$ *and* $\mathcal{L}_{i+1}$
23 $\mathbb{K}_{i+1} :=$ `induced-context`$(\mathcal{I}_\ell, M_{i+1})$
24 $\mathcal{K}_{i+1} := \{ P_r \to (P_r)''_{\mathbb{K}_{i+1}} \mid P_r \neq (P_r)''_{\mathbb{K}_{i+1}}, r \in \{ 0, \ldots, i \} \}$
25 $\mathcal{S}_{i+1} := \{ \{ A \} \to \{ B \} \mid A, B \in M_{i+1}, A \sqsubseteq B \}$
26 **choose** $\leqslant_{M_{i+1}}$ such that (7.1) and (7.2) hold
27
28 *;; next closed set*
29 $P_{i+1} :=$ `next-closure`$(M_{i+1}, \leqslant_{M_{i+1}}, P_i, \mathcal{K}_{i+1} \cup \mathcal{S}_{i+1})$
30 **if** $P_{i+1} =$ **nil exit**
31
32 $i := i + 1$
33 **end**
34
35 **return** $\{ \bigcap P \sqsubseteq (\bigcap P)^{\mathcal{I}_\ell \mathcal{I}_\ell} \mid (P \to P''_{\mathbb{K}_{i+1}}) \in \mathcal{K}_{i+1} \}$
36 **end**

is again proposed to the expert. If $\bigcap P_i \sqsubseteq (\bigcap P_i)^{\mathcal{I}_{\ell+1}\mathcal{I}_{\ell+1}}$, i. e. $\bigcap P_i \equiv (\bigcap P_i)^{\mathcal{I}_{\ell+1}\mathcal{I}_{\ell+1}}$, then the next set P_{i+1} is considered.

We are now able to adapt Algorithm 10 by replacing all references to the background interpretation by expert interactions. The result is shown in Algorithm 11. From our previous discussion we now easily obtain the following result.

7.1.9 Theorem (Theorem 6.16 from [41]) *Let $\mathcal{I}_{\mathsf{back}}$ be a finite interpretation, and let $\mathcal{I}$ be a connected subinterpretation of $\mathcal{I}_{\mathsf{back}}$. Then Algorithm 11 applied to $\mathcal{I}$, using $\mathcal{I}_{\mathsf{back}}$ as background interpretation, terminates after finitely many steps. If n is the number of iterations in this run, and if $\mathcal{I}_\ell$ is the final working interpretation, then the set*

$$\{\bigcap P \sqsubseteq (\bigcap P)^{\mathcal{I}_\ell\mathcal{I}_\ell} \mid (P \to P''_{\mathbb{K}_{n+1}}) \in \mathcal{K}_{n+1}\}$$

is a finite base of $\mathcal{I}_{\mathsf{back}}$.

7.2. Model Exploration with Confident GCIs

In the previous section we have seen how we can obtain an algorithm for model exploration by extending Baader and Distel's results on computing finite bases of finite interpretations. In this section we want to generalize this argumentation to the setting of GCIs with high confidence, i. e. we want to obtain an algorithm for model exploration which not only asks GCIs which are valid in the current working interpretation, but which is also allowed to ask GCIs whose confidence in the original data is just high enough. This process we shall call *model exploration by confidence*.

The argumentation used for this model exploration algorithm essentially consists of amending the computation of finite bases of finite interpretations by suitable expert interaction. Consequently, the argumentation we shall develop for model exploration by confidence will be based on the computation of bases of GCIs with high confidence. However, the interpretation we consider during the exploration process contains a connected subinterpretation consisting of the counterexamples given by the expert, and all GCIs which are not valid within this subinterpretation should not be considered further, even if they have a confidence above c in the initial working interpretation.

We can thus think of $\mathcal{I}_\ell$ as consisting of two parts: the initial working interpretation $\mathcal{I}$, which may contain errors and where we apply our confidence heuristics, and a connected subinterpretation $\mathcal{I}_\ell \backslash \mathcal{I}$, consisting of the counterexamples given by the expert, where we only consider valid GCIs. We can think of all elements of $\mathcal{I}$ as *untrusted*, and of all elements of $\mathcal{I}_\ell \backslash \mathcal{I}$ as *trusted*.

To generalize the argumentation for model exploration to GCIs with high confidence, we shall thus start by devising an algorithm that allows us to compute finite bases of finite interpretations containing trusted and untrusted elements, i. e. that computes bases of

$$\mathrm{Th}_c(\mathcal{I}) \cap \mathrm{Th}(\mathcal{I}_\ell \backslash \mathcal{I}).$$

We shall do this in Section 7.2.1.

Thereafter, we shall follow the argumentation of the previous section. This means that in Section 7.2.2 we shall discuss an algorithm that computes bases of formal contexts containing trusted and untrusted objects, and where the attribute set is allowed to grow during the computation. Thereafter, we shall discuss in Section 7.2.2 how we can adapt this algorithm to compute bases of finite interpretations that contain trusted and untrusted individuals, and where the set $M_{\mathcal{I}}$ is computed incrementally during the run of the algorithm. Finally, we shall see in Section 7.2.3 how we can introduce suitable expert interaction to obtain an algorithm for model exploration by confidence.

7.2.1. Bases of Finite Interpretations with Untrusted Elements

Let $\mathcal{J}$ be a finite interpretation over N_C and N_R, and let $\mathcal{I}$ be a subinterpretation of $\mathcal{J}$. As already discussed, we want to think of $\mathcal{I}$ as the interpretation of *untrusted* elements, and of the interpretation

$$\mathcal{J}\backslash\mathcal{I} := (\Delta^{\mathcal{J}}\backslash\Delta^{\mathcal{I}}, \cdot^{\mathcal{J}\backslash\mathcal{I}})$$

as the interpretation of *trusted* elements (provided by the expert), where we define

$$A^{\mathcal{J}\backslash\mathcal{I}} := A^{\mathcal{J}} \cap \Delta^{\mathcal{J}}\backslash\Delta^{\mathcal{I}} = A^{\mathcal{J}}\backslash\Delta^{\mathcal{I}},$$
$$r^{\mathcal{J}\backslash\mathcal{I}} := r^{\mathcal{J}} \cap (\Delta^{\mathcal{J}}\backslash\Delta^{\mathcal{I}}) \times (\Delta^{\mathcal{J}}\backslash\Delta^{\mathcal{I}})$$

for $A \in N_C$ and $r \in N_R$.

The aim of this section is to obtain a method to find finite bases of $\mathcal{J}$ with untrusted elements $\mathcal{I}$. More precisely, let us define for $c \in [0,1]$ the set

$$\mathrm{Th}_c(\mathcal{J},\mathcal{I}) := \{\, C \sqsubseteq D \mid C, D \in \mathcal{EL}^{\bot}_{\mathrm{gfp}}(N_C, N_R),\ C^{\mathcal{J}}\backslash\Delta^{\mathcal{I}} \subseteq D^{\mathcal{J}}\backslash\Delta^{\mathcal{I}}, |(C \sqcap D)^{\mathcal{J}} \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |C^{\mathcal{J}} \cap \Delta^{\mathcal{I}}| \,\}.$$

We then want to describe a finite base of $\mathrm{Th}_c(\mathcal{J},\mathcal{I})$.

The results of this section have been published previously in [29].

Note that we have given the confidence constraint in the form of

$$|(C \sqsubseteq D)^{\mathcal{J}} \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |C^{\mathcal{J}} \cap \Delta^{\mathcal{I}}|, \tag{7.3}$$

which is the suitable formulation for our setting of $\mathcal{I}$ being a subinterpretation of $\mathcal{J}$. On the other hand, in our later considerations, both $\mathcal{I}$ and $\mathcal{J}\backslash\mathcal{I}$ will be connected subinterpretations of $\mathcal{J}$, and in this case the definition of $\mathrm{Th}_c(\mathcal{J},\mathcal{I})$ can be simplified as follows: recall that in the case that $\mathcal{I}$ is a connected subinterpretation of $\mathcal{J}$, Lemma 7.1.6 yields that for all $C, D \in \mathcal{EL}^{\bot}_{\mathrm{gfp}}(N_C, N_R)$

$$C^{\mathcal{J}} \cap \Delta^{\mathcal{I}} = C^{\mathcal{I}},$$

$$(C \sqcap D)^{\mathcal{J}} \cap \Delta^{\mathcal{I}} = (C \sqcap D)^{\mathcal{I}}.$$

Thus, Equation (7.3) simplifies to

$$|(C \sqcap D)^{\mathcal{I}}| \geqslant c \cdot |C^{\mathcal{I}}|,$$

which is equivalent to $\mathrm{conf}_{\mathcal{I}}(C \sqsubseteq D) \geqslant c$. Furthermore, since $\mathcal{J}\backslash\mathcal{I}$ is a connected subinterpretation of $\mathcal{J}$, we obtain again by Lemma 7.1.6 that

$$C^{\mathcal{J}\backslash\mathcal{I}} = C^{\mathcal{J}} \cap (\Delta^{\mathcal{J}}\backslash\Delta^{\mathcal{I}}) = C^{\mathcal{J}}\backslash\Delta^{\mathcal{I}}$$

for all $C \in \mathcal{EL}^{\bot}_{\mathrm{gfp}}(N_C, N_R)$. Therefore, $C^{\mathcal{J}}\backslash\Delta^{\mathcal{I}} \subseteq D^{\mathcal{J}}\backslash\Delta^{\mathcal{I}}$ is equivalent to $C^{\mathcal{J}\backslash\mathcal{I}} \subseteq D^{\mathcal{J}\backslash\mathcal{I}}$, i. e. $(C \sqsubseteq D) \in \mathrm{Th}(\mathcal{J}\backslash\mathcal{I})$. Thus, the definition of $\mathrm{Th}_c(\mathcal{J},\mathcal{I})$ can be rewritten as

$$\begin{aligned}\mathrm{Th}_c(\mathcal{J},\mathcal{I}) &= \{\, C \sqsubseteq D \mid C, D \in \mathcal{EL}^{\bot}_{\mathrm{gfp}}(N_C, N_R), C^{\mathcal{J}}\backslash\Delta^{\mathcal{I}} \subseteq D^{\mathcal{J}}\backslash\Delta^{\mathcal{I}}, \mathrm{conf}_{\mathcal{I}}(C \sqsubseteq D) \geqslant c \,\} \\ &= \{\, (C \sqsubseteq D) \in \mathrm{Th}(\mathcal{J}\backslash\mathcal{I}) \mid \mathrm{conf}_{\mathcal{I}}(C \sqsubseteq D) \geqslant c \,\} \\ &= \mathrm{Th}(\mathcal{J}\backslash\mathcal{I}) \cap \mathrm{Th}_c(\mathcal{I}),\end{aligned}$$

which corresponds to our intention of finding a finite base of all GCIs which are valid in $\mathcal{J}\backslash\mathcal{I}$ and have high confidence in $\mathcal{I}$.

Let us return to the general case that $\mathcal{I}$ is just a subinterpretation of $\mathcal{J}$. To find a base for the set $\mathrm{Th}_c(\mathcal{J}\backslash\mathcal{I})$, we make use of the ideas we have already used to find finite confident bases of $\mathrm{Th}_c(\mathcal{I})$. More precisely, we first observe that

$$\mathrm{Th}(\mathcal{J}) \subseteq \mathrm{Th}_c(\mathcal{J},\mathcal{I}).$$

Since we can find bases of $\mathrm{Th}(\mathcal{J})$ using the results of Baader and Distel, we again concentrate on finding bases of the set $\mathrm{Th}_c(\mathcal{J},\mathcal{I})\backslash\mathrm{Th}(\mathcal{J})$. In other words, if $\mathcal{B}$ is a base of $\mathrm{Th}(\mathcal{J})$, then we seek a set $\mathcal{C} \subseteq \mathrm{Th}_c(\mathcal{J},\mathcal{I})\backslash\mathrm{Th}(\mathcal{J})$ which is complete for $\mathrm{Th}_c(\mathcal{J},\mathcal{I})\backslash\mathrm{Th}(\mathcal{J})$. In this case, $\mathcal{B} \cup \mathcal{C}$ is a base of $\mathrm{Th}_c(\mathcal{J},\mathcal{I})$.

To find such a set $\mathcal{C}$ we first observe that

$$(C \sqsubseteq D) \in \mathrm{Th}_c(\mathcal{J},\mathcal{I}) \iff (C^{\mathcal{J}\mathcal{J}} \sqsubseteq D^{\mathcal{J}\mathcal{J}}) \in \mathrm{Th}_c(\mathcal{J},\mathcal{I}).$$

This is because

$$\begin{aligned}C^{\mathcal{J}} &= C^{\mathcal{J}\mathcal{J}\mathcal{J}}, \\ (C \sqcap D)^{\mathcal{J}} &= (C \sqcap D)^{\mathcal{J}\mathcal{J}\mathcal{J}}\end{aligned}$$

is true, thus

$$C^{\mathcal{J}}\backslash\Delta^{\mathcal{I}} \subseteq D^{\mathcal{J}}\backslash\Delta^{\mathcal{I}} \iff C^{\mathcal{J}\mathcal{J}\mathcal{J}}\backslash\Delta^{\mathcal{I}} \subseteq D^{\mathcal{J}\mathcal{J}\mathcal{J}}\backslash\Delta^{\mathcal{I}},$$

and Equation (7.3) is true if and only if

$$|(C \sqsubseteq D)^{\mathcal{J}\mathcal{J}\mathcal{J}} \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |C^{\mathcal{J}\mathcal{J}\mathcal{J}} \cap \Delta^{\mathcal{I}}|.$$

If now $\mathcal{B}$ is a base of $\mathrm{Th}(\mathcal{J})$, then it is true that

$$\mathcal{B} \cup \{\, C^{\mathcal{J}\mathcal{J}} \sqsubseteq D^{\mathcal{J}\mathcal{J}} \,\} \models (C \sqsubseteq D).$$

This is because $\mathcal{B} \models (C \sqsubseteq C^{\mathcal{J}\mathcal{J}})$, since $C \sqsubseteq C^{\mathcal{J}\mathcal{J}}$ is valid in $\mathcal{J}$. Furthermore, $D^{\mathcal{J}\mathcal{J}} \sqsubseteq D$, and thus

$$\mathcal{B} \cup \{\, C^{\mathcal{J}\mathcal{J}} \sqsubseteq D^{\mathcal{J}\mathcal{J}} \,\} \models (C \sqsubseteq C^{\mathcal{J}\mathcal{J}} \sqsubseteq D^{\mathcal{J}\mathcal{J}} \sqsubseteq D).$$

Having these two considerations in mind we define

$$\mathrm{Conf}(\mathcal{J}, c, \mathcal{I}) := \{\, X^{\mathcal{J}} \sqsubseteq Y^{\mathcal{J}} \mid Y \subseteq X \subseteq \Delta^{\mathcal{J}}, (X^{\mathcal{J}} \sqsubseteq Y^{\mathcal{J}}) \in \mathrm{Th}_c(\mathcal{J}, \mathcal{I}) \,\}.$$

Since $\mathcal{J}$ is a finite interpretation, $\Delta^{\mathcal{J}}$ is finite. We therefore obtain the following result.

7.2.1 Theorem *Let $\mathcal{J}$ be a finite interpretation, let $\mathcal{I}$ be a subinterpretation of $\mathcal{J}$, and let $c \in [0,1]$. Then if $\mathcal{B}$ is a finite base of $\mathcal{J}$, then the set*

$$\mathcal{B} \cup \mathrm{Conf}(\mathcal{J}, c, \mathcal{I})$$

is a finite base of $\mathrm{Th}_c(\mathcal{J}, \mathcal{I})$.

This result already solves our initial problem of finding a finite base of $\mathrm{Th}_c(\mathcal{J}\backslash\mathcal{I})$. In the following, we want to extend this result in the direction of computing finite bases of $\mathrm{Th}_c(\mathcal{J}, \mathcal{I})$ by computing suitable bases in the corresponding induced contexts. This will be helpful later when we develop our algorithm for model exploration by confidence.

To this end, we first need to introduce some extra notation. Let $X \subseteq \Delta^{\mathcal{J}}$. Then we shall denote with $\mathbb{K}_{\mathcal{J}} \restriction_X$ the formal context whose set of objects is restricted to X, i. e.

$$\mathbb{K}_{\mathcal{J}} \restriction_X := (X, M_{\mathcal{J}}, \nabla),$$

where $(x, C) \in \nabla \iff x \in C^{\mathcal{J}}$ for $x \in X, C \in M_{\mathcal{J}}$ as before.

We can now formulate a result that allows to find bases of interpretations with untrusted elements from bases of corresponding induced contexts.

7.2.2 Theorem *Let $\mathcal{J}$ be a finite interpretation over N_C and N_R, and let $\mathcal{I}$ be a subinterpretation of $\mathcal{J}$. Let $c \in [0,1]$, and define*

$$\mathcal{T} := \mathrm{Th}_c(\mathbb{K}_{\mathcal{J}} \restriction_{\Delta^{\mathcal{I}}}) \cap \mathrm{Th}(\mathbb{K}_{\mathcal{J}} \restriction_{\Delta^{\mathcal{J}} \backslash \Delta^{\mathcal{I}}}).$$

Let $\mathcal{L} \subseteq \mathcal{T}$ be complete for $\mathcal{T}$. Then $\bigsqcap \mathcal{L} \subseteq \mathrm{Th}_c(\mathcal{J}, \mathcal{I})$ and $\bigsqcap \mathcal{L}$ is complete for $\mathrm{Th}_c(\mathcal{J}, \mathcal{I})$.

Proof We first show $\bigsqcap \mathcal{L} \subseteq \mathrm{Th}_c(\mathcal{J},\mathcal{I})$. For this we need to show that for each $(\bigsqcap X \sqsubseteq \bigsqcap Y) \in \bigsqcap \mathcal{L}$ it is true that

i. $|(\bigsqcap X \sqcap \bigsqcap Y)^{\mathcal{J}} \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |(\bigsqcap X)^{\mathcal{J}} \cap \Delta^{\mathcal{I}}|$, and

ii. $(\bigsqcap X)^{\mathcal{J}} \backslash \Delta^{\mathcal{I}} \subseteq (\bigsqcap Y)^{\mathcal{J}} \backslash \Delta^{\mathcal{I}}$.

For the first subclaim we observe that $\mathrm{conf}_{\mathbb{K}_{\mathcal{J}} \upharpoonright_{\Delta^{\mathcal{I}}}}(X \to Y) \geqslant c$, i. e.

$$|(X \cup Y)' \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |X' \cap \Delta^{\mathcal{I}}|.$$

Since $X' = (\bigsqcap X)^{\mathcal{J}}$ and $Y' = (\bigsqcap Y)^{\mathcal{J}}$ by Proposition 4.2.9, we obtain

$$|(\bigsqcap (X \cup Y))^{\mathcal{J}} \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |(\bigsqcap X)^{\mathcal{J}} \cap \Delta^{\mathcal{I}}|,$$

and since $\bigsqcap (X \cup Y) = \bigsqcap X \sqcap \bigsqcap Y$ we finally get

$$|(\bigsqcap X \sqcap \bigsqcap Y)^{\mathcal{J}} \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |(\bigsqcap X)^{\mathcal{J}} \cap \Delta^{\mathcal{I}}|,$$

as required.

For the second subclaim we observe that $X' \backslash \Delta^{\mathcal{I}} \subseteq Y' \backslash \Delta^{\mathcal{I}}$, because $X \to Y$ is valid in $\mathbb{K}_{\mathcal{J}} \upharpoonright_{\Delta^{\mathcal{J}} \backslash \Delta^{\mathcal{I}}}$. Since $X' = (\bigsqcap X)^{\mathcal{J}}$ and $Y' = (\bigsqcap Y)^{\mathcal{J}}$, we obtain

$$(\bigsqcap X)^{\mathcal{J}} \backslash \Delta^{\mathcal{I}} \subseteq (\bigsqcap Y)^{\mathcal{J}} \backslash \Delta^{\mathcal{I}},$$

as required.

We have thus shown that $\bigsqcap \mathcal{L} \subseteq \mathrm{Th}_c(\mathcal{J},\mathcal{I})$. We shall now consider the completeness of $\bigsqcap \mathcal{L}$ for $\mathrm{Th}_c(\mathcal{J},\mathcal{I})$.

To this end, we shall show the following two subclaims

i. $\bigsqcap \mathcal{L} \models (\bigsqcap U \sqsubseteq (\bigsqcap U)^{\mathcal{J}\mathcal{J}})$ for all $U \subseteq M_{\mathcal{J}}$, and

ii. $\bigsqcap \mathcal{L} \models \mathrm{Conf}(\mathcal{J}, c, \mathcal{I})$.

Since

$$\{ \bigsqcap U \sqsubseteq (\bigsqcap U)^{\mathcal{J}\mathcal{J}} \mid U \subseteq M_{\mathcal{J}} \}$$

is a base of $\mathcal{J}$, the completeness of $\bigsqcap \mathcal{L}$ for $\mathrm{Th}_c(\mathcal{J},\mathcal{I})$ then follows immediately from Theorem 7.2.1.

For the first subclaim let $U \subseteq M_{\mathcal{J}}$. Because

$$\mathrm{Th}(\mathbb{K}_{\mathcal{J}}) \subseteq \mathrm{Th}_c(\mathbb{K}_{\mathcal{J}} \upharpoonright_{\Delta^{\mathcal{I}}}) \cap \mathrm{Th}(\mathbb{K}_{\mathcal{J}} \upharpoonright_{\Delta^{\mathcal{J}} \backslash \Delta^{\mathcal{I}}})$$

it follows that $\mathcal{L}$ is complete for $\mathbb{K}_{\mathcal{J}}$. Therefore,

$$\mathcal{L} \models (U \to U'').$$

By Lemma 5.2.19 we obtain

$$\bigsqcap \mathcal{L} \models (\bigsqcap U \sqsubseteq \bigsqcap U''),$$

and since $\bigsqcap U'' \equiv (\bigsqcap U)^{\mathcal{J}\mathcal{J}}$, we obtain

$$\bigsqcap \mathcal{L} \models (\bigsqcap U \sqsubseteq (\bigsqcap U)^{\mathcal{J}\mathcal{J}})$$

as required.

For the second subclaim let $(X^{\mathcal{J}} \sqsubseteq Y^{\mathcal{J}}) \in \mathrm{Conf}(\mathcal{J}, c, \mathcal{I})$. Then by Proposition 4.2.9 it is true that

$$X^{\mathcal{J}} \equiv \bigsqcap X',\ Y^{\mathcal{J}} \equiv \bigsqcap Y'.$$

Therefore,

$$\bigsqcap \mathcal{L} \models (X^{\mathcal{J}} \sqsubseteq Y^{\mathcal{J}}) \iff \bigsqcap \mathcal{L} \models (\bigsqcap X' \sqsubseteq \bigsqcap Y'). \tag{7.4}$$

Therefore, to show $\bigsqcap \mathcal{L} \models (X^{\mathcal{J}} \sqsubseteq Y^{\mathcal{J}})$ it suffices to show $\mathcal{L} \models (X' \to Y')$.

Recall that since $(X^{\mathcal{J}} \sqsubseteq Y^{\mathcal{J}}) \in \mathrm{Conf}(\mathcal{J}, c, \mathcal{I})$, it is true that

$$|(X^{\mathcal{J}} \sqcap Y^{\mathcal{J}})^{\mathcal{J}} \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |X^{\mathcal{J}\mathcal{J}} \cap \Delta^{\mathcal{I}}|.$$

This implies

$$|(\bigsqcap (X' \cup Y'))^{\mathcal{J}} \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |(\bigsqcap X')^{\mathcal{J}} \cap \Delta^{\mathcal{I}}|.$$

and thus

$$|((X' \cup Y')' \cap \Delta^{\mathcal{I}}| \geqslant c \cdot |X'' \cap \Delta^{\mathcal{I}}|,$$

i. e. $(X' \to Y') \in \mathrm{Th}_c(\mathbb{K}_{\mathcal{J}} \restriction_{\Delta^{\mathcal{I}}})$.

Furthermore, it is true that

$$X^{\mathcal{J}\mathcal{J}} \backslash \Delta^{\mathcal{I}} \subseteq Y^{\mathcal{J}\mathcal{J}} \backslash \Delta^{\mathcal{I}},$$

and by Proposition 4.2.9 we have $X^{\mathcal{J}\mathcal{J}} = X''$, $Y^{\mathcal{J}\mathcal{J}} = Y''$, thus

$$X'' \backslash \Delta^{\mathcal{I}} \subseteq Y'' \backslash \Delta^{\mathcal{I}},$$

i. e. $(X' \to Y') \in \mathrm{Th}(\mathbb{K}_{\mathcal{J}} \restriction_{\Delta^{\mathcal{J}} \backslash \Delta^{\mathcal{I}}})$.

Since $\mathcal{L}$ is complete for $\mathcal{T}$, we thus obtain that

$$\mathcal{L} \models (X' \to Y'),$$

and thus $\bigsqcap \mathcal{L} \models (\bigsqcap X' \sqsubseteq \bigsqcap Y')$ by Lemma 5.2.19, and $\bigsqcap \mathcal{L} \models (X^{\mathcal{J}} \sqsubseteq Y^{\mathcal{J}})$ by Equation (7.4). ▫

7.2.2. Computing Bases of Formal Contexts with Growing Sets of Attributes

We have seen how we can obtain finite bases of interpretations containing untrusted elements. In the following two sections we want to devise an algorithm that allows us to compute this base in a manner which is suitable for being adapted towards model exploration by confidence. In particular, we shall see in this section how we can compute bases of

$$\mathrm{Th}_c(\mathbb{K}_{\mathcal{J}} \restriction_{\Delta^{\mathcal{I}}}) \cap \mathrm{Th}(\mathbb{K}_{\mathcal{J}} \restriction_{\Delta^{\mathcal{J}} \setminus \Delta^{\mathcal{I}}}), \tag{7.5}$$

where we compute the set $M_{\mathcal{J}}$ incrementally during the run of the algorithm. Then, in the next section we shall see how we can use this algorithm and Theorem 7.2.2 to compute finite bases of interpretations that contain untrusted elements.

Let us consider the problem of finding an algorithm that allows us to compute bases of the set given in Equation (7.5) from a more abstract point of view. More precisely, let us consider two formal contexts $\mathbb{K}_1$ and $\mathbb{K}_2$ with the same attribute set M. Then we want to find an algorithm that computes a base of

$$\mathrm{Th}_c(\mathbb{K}_1) \cap \mathrm{Th}(\mathbb{K}_2), \tag{7.6}$$

and which allows us to incrementally supply the elements of M as the computation proceeds.

As a special case of Equation (7.6) we first consider the case that $\mathbb{K}_2 = (\varnothing, M, \varnothing)$, i. e. we want to devise the algorithm such that it computes a base of $\mathrm{Th}_c(\mathbb{K}_1)$. As in Section 7.1.1, we want to obtain such an algorithm by adapting the classical algorithm for computing the canonical base of a formal context. Indeed, we could simply obtain such an algorithm if we would replace in Algorithm 9 every occurrence of $(\cdot)''_{\mathbb{K}_{i+1}}$ by a call to the closure operator induced by $\mathrm{Th}_c(\mathbb{K}_{i+1})$. However, as we had already argued in Section 6.2.1, computing closures under $\mathrm{Th}_c(\mathbb{K}_{i+1})$ may be infeasible.

To avoid this, we shall make use of the ideas we have developed in Section 6.2.2, when we devised an algorithm for exploration by confidence that avoids computing closures under $\mathrm{Th}_c(\mathbb{K})$. Indeed, we can just take Algorithm 8, and instantiate it with an expert that confirms all implications.

Recall that in this algorithm there were two cases of implications asked to the expert: implications were either of the form $P_{i+1} \to \{ m \}$, where $\mathrm{conf}_{\mathbb{K}}(P_{i+1} \to \{ m \}) \geqslant c$ and $c \notin \mathcal{K}_i(P_{i+1})$, or $P_{i+1} \to (P_{i+1})''_{\mathbb{K} \div \mathbb{L}_i}$. We can simplify these two cases into one case by defining for $P \subseteq M$ and $c \in [0, 1]$

$$P^{\mathbb{K},c} := \{ m \in M \mid \mathrm{conf}_{\mathbb{K}}(P \to \{ m \}) \geqslant c \}.$$

Then we only ask implications of the form

$$P_{i+1} \to (P_{i+1})^{\mathbb{K},c},$$

Algorithm 12 Axiomatize Confident Implications with Growing Sets of Attributes

define `confident-base`($\mathbb{K} = (G, M, I), c \in [0,1]$)
  $i := 0$
  $M_i := M$
  $I_i := I$
  $\mathcal{S}_i := P_i := \mathcal{K}_i := \varnothing$
  choose $\leqslant_{M_i}$ as a linear order on M_i

  forever do
    read $\mathbb{K}_{i+1} = (G, M_{i+1}, I_{i+1})$ such that $M_i \subseteq M_{i+1}$ and $I_i = (G \times M_i) \cap I_{i+1}$
    read $\mathcal{S}_{i+1}$ such that $\mathcal{S}_i \subseteq \mathcal{S}_{i+1} \subseteq \mathrm{Th}_c(\mathbb{K}_{i+1})$
    choose $\leqslant_{M_{i+1}}$ such that (7.1) and (7.2) hold

    $\mathcal{K}_{i+1} := \{\, P_k \to P_k^{\mathbb{K}_{i+1},c} \mid k \in \{0,\dots,i\}, P_k \neq P_k^{\mathbb{K}_{i+1},c} \,\}$

    $P_{i+1}^1 := \texttt{next-closure}(M_{i+1}, \leqslant_{M_{i+1}}, P_i, \mathbb{K}_{i+1})$
    $P_{i+1}^2 := \texttt{next-closure}(M_{i+1}, \leqslant_{M_{i+1}}, P_i, \mathcal{S}_{i+1} \cup \mathcal{K}_{i+1})$

    $P_{i+1} := \min_{\leq}(P_{i+1}^1, P_{i+1}^2)$.
    if $P_{i+1} =$ **nil exit**

    $i := i + 1$
  end

  return $\mathcal{K}_{i+1}$
end

and this then covers both cases.

To make this algorithm into an algorithm that allows the set of attributes to grow during the computation, we use the ideas of Section 7.1.1: whenever there are new elements to be added to the current set of attributes, we add them as the smallest elements. In this way, the underlying Next Closure algorithm behaves as if those elements would have been present right from the start of the run, and thus behaves as desired.

The algorithm that we obtain from these considerations is shown in Algorithm 12. Note that as in the case of Algorithm 9, we cannot discard sets P_i which are closed under $(\cdot)^{\mathbb{K}_{i+1},c}$, i. e. which satisfy $P_i = P_i^{\mathbb{K}_{i+1},c}$, as P_i may not be closed under $(\cdot)^{\mathbb{K}_{j},c}$ for some later iteration j.

We can argue termination of Algorithm 12 as we did before for Algorithm 9: if at a

certain iteration ℓ it is true for all iterations $k \geqslant \ell$ that $M_k = M_\ell$, then Algorithm 12 must terminate. This is in particular the case if we want to compute a base of $\mathrm{Th}_c(\mathbb{K})$ where $\mathbb{K}$ is a finite formal context, and where the attributes of $\mathbb{K}$ are supplied incrementally during the run of the algorithm.

To show that upon termination, the set $\mathcal{K}_n$ of implications, where n is the number of iterations of Algorithm 12, is indeed a base of $\mathrm{Th}_c(\mathbb{K}_n)$, we adapt the argumentation of Section 7.1.1 and Section 7.1.2 accordingly. The following result and its proof are a generalization of [41, Lemma 6.3].

7.2.3 Proposition *Consider a terminating run of Algorithm 12, and let n be the number of iterations of this run. Let $Q \subseteq M_n$. Then the following statements hold.*

i. If $Q \neq Q''_{\mathbb{K}_n}$, then Q is not $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed.

ii. If $Q = Q''_{\mathbb{K}_n}$, then $Q = P_k$ for some $k \in \{ 0, \ldots, n \}$.

Proof The case $Q = \varnothing = P_0$ can be handled quite easily: if $Q \neq Q''_{\mathbb{K}_n}$, then $Q \neq Q^{\mathbb{K}_n, c}$, since $Q''_{\mathbb{K}_n} \subseteq Q^{\mathbb{K}_n, c}$. Therefore, $(Q \to Q^{\mathbb{K}_n, c}) \in \mathcal{K}_n$ and thus Q is not $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed. If on the other hand $Q = Q''_{\mathbb{K}_n}$, then $Q = P_0$ shows the claim.

Now suppose that $Q \neq \varnothing$. Then there exists $k \in \{ 0, \ldots, n \}$ such that

$$P_{k-1} \precneqq Q \preceq P_k.$$

We first argue that $Q \subseteq M_k$. To this end, suppose by contradiction that this is not the case, and let $m \in Q \setminus M_k$. Since $m \notin M_k$, it is smaller than every element of M_k, by construction of the linear order $\leqslant_{M_n}$ on M_n. Thus, $M_k \precneqq \{ m \}$, and since $\{ m \} \subseteq Q$, we obtain

$$M_k \precneqq \{ m \} \preceq Q,$$

contradicting the fact that $Q \preceq P_k \preceq M_k$. Therefore, $Q \subseteq M_k$.

Let us first consider the case $Q \neq Q''_{\mathbb{K}_n}$, and assume by contradiction that Q is $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed. Then by construction

$$P_k^2 \preceq Q \preceq P_k \preceq P_k^2,$$

and thus $Q = P_k$. Then $Q \neq Q''_{\mathbb{K}_n}$ means $P_k \neq (P_k)''_{\mathbb{K}_n}$, thus $P_k \neq P_k^{\mathbb{K}_n, c}$, and therefore

$$(P_k \to P_k^{\mathbb{K}_n, c}) \in \mathcal{K}_n.$$

But Q is not $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed, a contradiction. Therefore, Q is not $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed, as it was claimed.

Let us now consider the case that $Q = Q''_{\mathbb{K}_n}$, and we have to show that $Q = P_\ell$ for $\ell \in \{ 0, \ldots, n \}$. Since $Q = Q''_{\mathbb{K}_n}$, it is also true that $Q = Q''_{\mathbb{K}_k}$, since

$$I_k = (G \times M_k) \cap I_n.$$

But then $P_k^1 \preceq Q \preceq P_k$, and thus $Q = P_k$, as required. □

7.2.4 Theorem *Let $\mathbb{K}$ be a finite formal context, $c \in [0,1]$, and suppose that Algorithm 12 applied to $\mathbb{K}$ and c terminates after n iterations. Then $\mathcal{K}_n$ is a base for $\mathrm{Th}_c(\mathbb{K}_n)$ with background knowledge $\mathcal{S}_n$.*

Proof The fact that $\mathcal{K}_n \cup \mathcal{S}_n \subseteq \mathrm{Cn}(\mathrm{Th}_c(\mathbb{K}_n))$ is clear from the definition of $\mathcal{K}_n$ and $\mathcal{S}_n$, and we thus only need to show that $\mathcal{S}_n \cup \mathcal{K}_n$ is complete for $\mathrm{Th}_c(\mathbb{K}_n)$. For this we shall use Lemma 2.4.2 and show that every set $Q \subseteq M_n$ which is $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed is also $\mathrm{Th}_c(\mathbb{K}_n)$-closed.

To this end, let us assume by contradiction that Q is $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed, but not $\mathrm{Th}_c(\mathbb{K}_n)$-closed. Then there exists an implication $(P \to \{m\}) \in \mathrm{Th}_c(\mathbb{K}_n)$ such that $P \subseteq Q$ and $m \notin Q$. Furthermore, since Q is $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed, it follows from Proposition 7.2.3 that $Q = Q''_{\mathbb{K}_n}$. Then since

$$\mathrm{conf}_{\mathbb{K}_n}(P \to \{m\}) = \mathrm{conf}_{\mathbb{K}_n}(P''_{\mathbb{K}_n} \to \{m\}),$$

we can assume that $P = P''_{\mathbb{K}_n}$. But then, using Proposition 7.2.3 again, we obtain that $P = P_k$ for some $k \in \{0, \ldots, n\}$, and thus

$$(P \to P^{\mathbb{K}_n,c}) = (P_k \to P_k^{\mathbb{K}_n,c}) \in \mathcal{K}_n,$$

because $P_k \neq P_k^{\mathbb{K}_n,c}$, since $m \notin P_k \subseteq Q$, but $m \in P_k^{\mathbb{K}_n,c}$. Now since Q is $\mathcal{K}_n$-closed, $P_k \subseteq Q$ implies $P_k^{\mathbb{K}_n,c} \subseteq Q$, and since $m \in P_k^{\mathbb{K}_n,c}$, we obtain $m \in Q$, a contradiction.

Therefore, every set Q that is $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed is also $\mathrm{Th}_c(\mathbb{K}_n)$-closed, and thus $\mathcal{S}_n \cup \mathcal{K}_n$ is complete for $\mathrm{Th}_c(\mathbb{K}_n)$, as it was claimed. □

In this theorem, the set $\mathcal{K}_n$ does not necessarily contain implications whose confidence is at least c, i. e. $\mathcal{K}_n \subseteq \mathrm{Th}_c(\mathbb{K})$ is not necessarily true. However, a simple modification of Algorithm 12 achieves that the computed base is indeed a confident base of $\mathrm{Th}_c(\mathbb{K})$. For this we define the return value of Algorithm 12 as

$$\hat{\mathcal{K}} := \{ P \to \{m\} \mid (P \to P^{\mathbb{K},c}) \in \mathcal{K}_n, m \in P^{\mathbb{K},c} \}.$$

Then, by definition of $P^{\mathbb{K},c}$, it is true that $\hat{\mathcal{K}} \subseteq \mathrm{Th}_c(\mathbb{K})$. Of course, instead of choosing m in $P^{\mathbb{K},c}$, it also suffices to consider only $m \in P^{\mathbb{K},c} \setminus \mathcal{S}_n(P)$.

With Algorithm 12 we have now obtained an algorithm that allows us to compute bases of $\mathrm{Th}_c(\mathbb{K})$, where the attribute set can be added incrementally during the computation. Based on this algorithm, we now want to turn back to our initial problem of finding bases of $\mathrm{Th}_c(\mathbb{K}_1) \cap \mathrm{Th}(\mathbb{K}_2)$, where both formal contexts $\mathbb{K}_1$ and $\mathbb{K}_2$ have the same attribute set M.

The main idea to adapt Algorithm 12 to this setting is to *divide* the working context into two formal contexts $\mathbb{K}_{i+1}$ and $\mathbb{L}_{i+1}$, such that in $\mathbb{K}_{i+1}$ (the *untrusted* part) we apply the usual

confidence heuristics, and in $\mathbb{L}_{i+1}$ (the *trusted* part) we consider only valid implications. Then instead of computing the sets $P^{\mathbb{K}_{i+1},c}$, we consider

$$P^{\mathbb{K}_{i+1},c} \cap P''_{\mathbb{L}_{i+1}} = \{\, m \in P''_{\mathbb{L}_{i+1}} \mid \mathrm{conf}_{\mathbb{K}_{i+1}}(P \to \{\, m \,\}) \geqslant c \,\}.$$

The division of the working context into $\mathbb{K}_{i+1}$ and $\mathbb{L}_{i+1}$ can be represented by using the subposition $\mathbb{K}_{i+1} \div \mathbb{L}_{i+1}$ as the working context. Then clearly

$$P''_{\mathbb{K}_{i+1} \div \mathbb{L}_{i+1}} = P''_{\mathbb{K}_{i+1}} \cap P''_{\mathbb{L}_{i+1}} \subseteq P^{\mathbb{K}_{i+1},c} \cap P''_{\mathbb{L}_{i+1}}. \tag{7.7}$$

The resulting algorithm is shown in Algorithm 13.

Because of Equation (7.7) the proofs of Proposition 7.2.3 and Theorem 7.2.4 can be carried over to Algorithm 13 almost literally, essentially by replacing every occurrence of $(\cdot)''_{\mathbb{K}_i}$ by $(\cdot)''_{\mathbb{K}_i \div \mathbb{L}_i}$, and by replacing every expression of the form $P^{\mathbb{K}_i,c}$ by $P^{\mathbb{K}_i,c} \cap P''_{\mathbb{L}_i}$. From this we obtain the validity of the following results.

7.2.5 Proposition *Consider a terminating run of Algorithm 13, and let n be the number of iterations of this run. Let $Q \subseteq M_n$. Then the following statements hold.*

i. If $Q \neq Q''_{\mathbb{K}_n \div \mathbb{L}_n}$, then Q is not $(\mathcal{S}_n \cup \mathcal{K}_n)$-closed.

ii. If $Q = Q''_{\mathbb{K}_n \div \mathbb{L}_n}$, then $Q = P_k$ for some $k \in \{\, 0, \ldots, n \,\}$.

7.2.6 Theorem *Let $\mathbb{K}, \mathbb{L}$ be two finite formal contexts with attribute set M and disjoint sets of objects, $c \in [0, 1]$, and suppose that Algorithm 13 applied to $\mathbb{K}, \mathbb{L}$ and c terminates after n iterations. Then $\mathcal{K}_n$ is a base for $\mathrm{Th}_c(\mathbb{K}_n) \cap \mathrm{Th}(\mathbb{L}_n)$ with background knowledge $\mathcal{S}_n$.*

7.2.3. Computing Bases of Finite Interpretations with Untrusted Elements

We shall now use the results of the previous section to devise an algorithm that allows us to compute bases of interpretations with untrusted elements. More precisely, let $\mathcal{J}$ be a finite interpretation over N_C and N_R, and let $\mathcal{I}$ be a finite, connected subinterpretation of $\mathcal{J}$, such that $\mathcal{J} \backslash \mathcal{I}$ is also a connected subinterpretation of $\mathcal{J}$. We then want to adapt Algorithm 13 to compute a finite base of $\mathrm{Th}_c(\mathcal{J}, \mathcal{I})$.

To this end, we consider as input for Algorithm 13 the induced formal context $\mathbb{K}_{\mathcal{J}}$, represented as the subposition of $\mathbb{K}_{\mathcal{J}} \restriction_{\Delta^{\mathcal{I}}}$ and $\mathbb{K}_{\mathcal{J}} \restriction_{\Delta^{\mathcal{J}} \setminus \Delta^{\mathcal{I}}}$, where the common attribute set $M_{\mathcal{I}}$ is computed incrementally as in Algorithm 10. The result is shown in Algorithm 14. Note that in this algorithm we again utilize the idea of using

$$\mathcal{S}_{i+1} = \{\, \{\, C \,\} \to \{\, D \,\} \mid C, D \in M_{i+1}, C \sqsubseteq D \,\}$$

as background knowledge, because the resulting set $\bigsqcap \mathcal{S}_{i+1}$ of GCIs is trivial, but the set $\mathcal{S}_{i+1}$ of implications is not.

Algorithm 13 Axiomatize Confident Implications with Trusted Objects

define `confident-base/trusted-objects`($\mathbb{K} = (G_1, M, I), \mathbb{L} = (G_2, M, J), c \in [0,1]$)
  $i := 0$
  $M_i := M$
  $I_i := I$
  $J_i := J$
  $\mathcal{S}_i := P_i := \mathcal{K}_i := \varnothing$
  choose $\leqslant_{M_i}$ as a linear order on M_i

  forever do
    read M_{i+1} such that $M_i \subseteq M_{i+1}$
    read I_{i+1} such that $I_i = (G_1 \times M_i) \cap I_{i+1}$
    read J_{i+1} such that $J_i = (G_2 \times M_i) \cap J_{i+1}$
    $\mathbb{K}_{i+1} := (G_1, M_{i+1}, I_{i+1})$
    $\mathbb{L}_{i+1} := (G_2, M_{i+1}, J_{i+1})$
    read $\mathcal{S}_{i+1}$ such that $\mathcal{S}_i \subseteq \mathcal{S}_{i+1} \subseteq \mathrm{Th}_c(\mathbb{K}_{i+1}) \cap \mathrm{Th}(\mathbb{L}_{i+1})$
    choose $\leqslant_{M_{i+1}}$ such that (7.1) and (7.2) hold

    $\mathcal{K}_{i+1} := \{\, P_k \to P_k^{\mathbb{K}_{i+1},\mathcal{C}} \cap P''_{\mathbb{L}_{i+1}} \mid k \in \{0, \ldots, i\}, P_k \neq P_k^{\mathbb{K}_{i+1},\mathcal{C}} \cap P''_{\mathbb{L}_{i+1}} \,\}$

    $P^1_{i+1} :=$ `next-closure`$(M_{i+1}, \leqslant_{M_{i+1}}, P_i, \mathbb{K}_{i+1} \div \mathbb{L}_{i+1})$
    $P^2_{i+1} :=$ `next-closure`$(M_{i+1}, \leqslant_{M_{i+1}}, P_i, \mathcal{S}_{i+1} \cup \mathcal{K}_{i+1})$

    $P_{i+1} := \min_{\leq}(P^1_{i+1}, P^2_{i+1})$.
    if $P_{i+1} =$ **nil exit**

    $i := i + 1$
  end

  return $\mathcal{K}_{i+1}$
end

Algorithm 14 Axiomatize Confident GCIs in the Presence of Trusted Elements

define `confident-base-gcis/trusted-elements`$(\mathcal{J}, \mathcal{I}, c \in [0,1])$
  $i := 0$
  $M_i := N_C \cup \{ \bot \}$
  $\mathcal{S}_i := \{ \{ \bot \} \to \{ A \} \mid A \in N_C \}$
  $P_i := \mathcal{K}_i := \varnothing$
  choose $\leqslant_{M_i}$ as a linear order on M_i

  forever do
    $M_{i+1} := M_i \cup \{ \exists r.(\bigsqcap P_i)^{\mathcal{J}\mathcal{J}} \mid r \in N_R \}$ *;; union up to equivalence*
    $\mathbb{K}_{i+1} :=$ `induced-context`$(\mathcal{I}, M_{i+1})$
    $\mathbb{L}_{i+1} :=$ `induced-context`$(\mathcal{J}\backslash\mathcal{I}, M_{i+1})$
    $\mathcal{S}_{i+1} := \{ \{ C \} \to \{ D \} \mid C, D \in M_{i+1}, C \sqsubseteq D \}$.
    choose $\leqslant_{M_{i+1}}$ such that (7.1) and (7.2) hold

    $\mathcal{K}_{i+1} := \{ P_k \to P_k^{\mathbb{K}_{i+1},c} \cap P''_{\mathbb{L}_{i+1}} \mid k \in \{0, \ldots, i\}, P_k \neq P_k^{\mathbb{K}_{i+1},c} \cap P''_{\mathbb{L}_{i+1}} \}$

    $P^1_{i+1} :=$ `next-closure`$(M_{i+1}, \leqslant_{M_{i+1}}, P_i, \mathbb{K}_{i+1} \div \mathbb{L}_{i+1})$
    $P^2_{i+1} :=$ `next-closure`$(M_{i+1}, \leqslant_{M_{i+1}}, P_i, \mathcal{S}_{i+1} \cup \mathcal{K}_{i+1})$

    $P_{i+1} := \min_{\preceq}(P^1_{i+1}, P^2_{i+1})$.
    if $P_{i+1} =$ **nil exit**

    $i := i + 1$
  end

  return $\mathcal{K}_{i+1}$
end

To see that Algorithm 14 indeed computes a base of $\mathrm{Th}_c(\mathcal{J},\mathcal{I})$, we shall first argue that it is of the form of Algorithm 13. Thereafter, we shall show that when Algorithm 14 is applied to $\mathcal{J}$, $\mathcal{I}$, and $c \in [0,1]$, and terminates after n iterations, then $M_n = M_{\mathcal{J}}$ up to equivalence. Thus, by Theorem 7.2.6, the algorithm computes a base $\mathcal{K}_n$ of $\mathrm{Th}_c(\mathbb{K}_{\mathcal{J}}{\restriction_{\Delta^{\mathcal{I}}}}) \cap \mathrm{Th}(\mathbb{K}_{\mathcal{J}}{\restriction_{\Delta^{\mathcal{J}}\setminus\Delta^{\mathcal{I}}}})$, and then Theorem 7.2.2 yields that $\bigcap \mathcal{K}_n$ is a base of $\mathrm{Th}_c(\mathcal{J},\mathcal{I})$.

To argue that Algorithm 14 is of the form of Algorithm 13 we need to show that the variables $M_{i+1}, \mathbb{K}_{i+1}, \mathbb{L}_{i+1}, \mathcal{S}_{i+1}$ computed in Algorithm 14 satisfy the constraints given in Algorithm 13. However, this is quite clear from the definition of these variables: it is obvious that $M_i \subseteq M_{i+1}$, and that the incidence relations of $\mathbb{K}_{i+1}$ and $\mathbb{L}_{i+1}$ restricted to M_i are the incidence relations of $\mathbb{K}_i$ and $\mathbb{L}_i$, respectively. Furthermore, $\mathcal{S}_{i+1} \subseteq \mathrm{Th}_c(\mathbb{K}_{i+1}) \cap \mathrm{Th}(\mathbb{L}_{i+1})$, because $\mathcal{S}_{i+1}$ is even valid in $\mathbb{K}_{i+1} \div \mathbb{L}_{i+1}$.

Note that since $\mathcal{J}$ is a finite interpretation, the set $M_{\mathcal{J}}$ is finite. Since $M_i \subseteq M_{\mathcal{J}}$ holds for all iterations i, up to equivalence, it is true that from a certain iteration ℓ on, $M_\ell = M_k$ is true for all $k \geqslant \ell$. Thus, Algorithm 14 terminates on input $\mathcal{J}$, $\mathcal{I}$, and c.

To show that upon termination of Algorithm 14, $M_n = M_{\mathcal{J}}$ is true up to equivalence, we start with the following result, which is an adaption of [41, Lemma 6.7].

7.2.7 Proposition *Consider a terminating run of Algorithm 14, and let n be the number of iterations. Then for every $U \subseteq M_n$ and $r \in N_R$ it is true that*

$$\exists r.(\textstyle\bigcap U)^{\mathcal{J}\mathcal{J}} \in M_n$$

up to equivalence.

Proof Note that because both $\mathcal{I}$ and $\mathcal{J}\setminus\mathcal{I}$ are connected subinterpretations of $\mathcal{J}$, Lemma 7.1.6 shows that $\mathbb{K}_n \div \mathbb{L}_n$ is indeed the induced context of $\mathcal{J}$ and M_n. Thus, we obtain from Proposition 4.2.9 that

$$(\textstyle\bigcap U''_{\mathbb{K}_n \div \mathbb{L}_n})^{\mathcal{J}} = U'''_{\mathbb{K}_n \div \mathbb{L}_n} = U'_{\mathbb{K}_n \div \mathbb{L}_n} = (\bigcap U)^{\mathcal{J}}.$$

Therefore, it is true that

$$\exists r.(\textstyle\bigcap U''_{\mathbb{K}_n \div \mathbb{L}_n})^{\mathcal{J}\mathcal{J}} \equiv \exists r.(\bigcap U)^{\mathcal{J}\mathcal{J}}.$$

Since Algorithm 14 is a special case of Algorithm 13, Proposition 7.2.3 is also applicable to Algorithm 14, and we thus obtain a $k \in \{0,\ldots,n\}$ such that $U''_{\mathbb{K}_n \div \mathbb{L}_n} = P_k$. Since $\exists r.(\bigcap P_k)^{\mathcal{J}\mathcal{J}} \in M_{k+1} \subseteq M_n$, we obtain

$$\exists r.(\textstyle\bigcap U)^{\mathcal{J}\mathcal{J}} \equiv \exists r.(\bigcap U''_{\mathbb{K}_n \div \mathbb{L}_n})^{\mathcal{J}\mathcal{J}} \equiv \exists r.(\bigcap P_k)^{\mathcal{J}\mathcal{J}} \in M_n,$$

as desired. □

Using this proposition, we shall now show the correctness of Algorithm 14.

7.2.8 Theorem *Let $\mathcal{J}$ be a finite interpretation over N_C and N_R, and let $\mathcal{I}$ be a connected subinterpretation of $\mathcal{J}$ such that $\mathcal{J}\backslash\mathcal{I}$ is also a connected subinterpretation of $\mathcal{J}$. Let $c \in [0,1]$, and let n be the number of iterations of Algorithm 14 when applied to $\mathcal{J}, \mathcal{I}$, and $c \in [0,1]$. Then $\bigsqcap \mathcal{K}_n$ is a base of* $\mathrm{Th}_c(\mathcal{J}, \mathcal{I})$.

The proof of this theorem, which is an adaption of the proofs of [41, Lemma 6.8, Theorem 6.9], uses induction over the role depth of concept descriptions. Since $\bigsqcap \mathcal{K}_n$ may contain proper $\mathcal{EL}^{\bot}_{\mathrm{gfp}}$ concept descriptions, it may not be immediately obvious how this can be done, and we need an extra result that allows us to use this argumentation.

7.2.9 Lemma (Lemma 5.6 from [41]) *Let $\mathcal{I}$ be a finite interpretation over N_C and N_R, and let C be an $\mathcal{EL}^{\bot}_{gfp}$ concept description over N_C and N_R. Then there exists an $\mathcal{EL}^{\bot}$ concept description over N_C and N_R such that*

$$C^{\mathcal{I}} = D^{\mathcal{I}} \quad \textit{and} \quad C \sqsubseteq D.$$

We now prove Theorem 7.2.8.

Proof (Theorem 7.2.8) We first show that $M_n = M_{\mathcal{J}}$ is true up to equivalence, i. e. every element of M_n is equivalent to some element in $M_{\mathcal{J}}$ and vice versa. Since $M_n \subseteq M_{\mathcal{J}}$ up to equivalence by definition of M_n, it suffices to show that every element of $M_{\mathcal{J}}$ is equivalent to some element in M_n.

To show that $M_{\mathcal{J}} \subseteq M_n$ holds up to equivalence, we shall show that for each $r \in N_R$ and $X \subseteq \Delta^{\mathcal{J}}$ there exists $C \in M_n$ such that $C \equiv \exists r.X^{\mathcal{J}}$. To this end, we observe that by Lemma 7.2.9 there exists an $\mathcal{EL}^{\bot}$ concept description D over N_C and N_R such that

$$D^{\mathcal{J}} = X^{\mathcal{J}\mathcal{J}}.$$

Since $D^{\mathcal{J}\mathcal{J}} = X^{\mathcal{J}\mathcal{J}\mathcal{J}} = X^{\mathcal{J}}$, it is sufficient to show that for each $\mathcal{EL}^{\bot}$ concept description D and each $r \in N_R$ it is true that

$$\exists r.D^{\mathcal{J}\mathcal{J}} \in M_n$$

up to equivalence. We shall show this claim by induction over the role-depth of D.

The *base case* is $D = \bot$, or D being a conjunction of concept names from N_C. The case $D = \bot$ is trivial, as $\exists r.\bot^{\mathcal{J}\mathcal{J}} \equiv \bot \in M_n$ for all $r \in N_R$. If $D = \bigsqcap S$ for some $S \subseteq N_C$, then since $S \subseteq M_n$, Proposition 7.2.7 implies for all $r \in N_R$ that

$$\exists r.D^{\mathcal{J}\mathcal{J}} = \exists r.(\textstyle\bigsqcap S)^{\mathcal{J}\mathcal{J}} \in M_n$$

up to equivalence.

For the *step case* let D be an $\mathcal{EL}^{\bot}$ concept description with role-depth $d > 0$, and let $r \in N_R$. Assume by induction for all $\mathcal{EL}^{\bot}$ concept descriptions E over N_C and N_R with role depth smaller than d that

$$\exists s.E^{\mathcal{J}\mathcal{J}} \in M_n \tag{7.8}$$

is true up to equivalence for all $s \in N_R$.

Since D is an $\mathcal{EL}^\bot$ concept description, there exist $U \subseteq N_C$, $r_1, \ldots, r_k \in N_R$ and $E_1, \ldots, E_k \in \mathcal{EL}^\bot(N_C, N_R)$ such that

$$D \equiv \bigsqcap U \sqcap \bigsqcap_{i=1}^{k} \exists r_i.E_i.$$

Then by Proposition 4.2.4

$$\begin{aligned} D^{\mathcal{J}\mathcal{J}} &\equiv \big(\bigsqcap U \sqcap \bigsqcap_{i=1}^{k} \exists r_i.E_i\big)^{\mathcal{J}\mathcal{J}} \\ &\equiv \big(\bigsqcap U \sqcap \bigsqcap_{i=1}^{k} \exists r_i.E_i^{\mathcal{J}\mathcal{J}}\big)^{\mathcal{J}\mathcal{J}}. \end{aligned}$$

By induction hypothesis Equation (7.8), $\exists r_i.E_i^{\mathcal{J}\mathcal{J}} \in M_n$ up to equivalence, for all $i = 1, \ldots, k$. But then

$$V := U \cup \{\, \exists r_i.E_i^{\mathcal{J}\mathcal{J}} \mid i = 1, \ldots, k \,\} \subseteq M_n,$$

and Proposition 7.2.7 implies that

$$\exists r.D^{\mathcal{J}\mathcal{J}} \equiv \exists r.\big(\bigsqcap V\big)^{\mathcal{J}\mathcal{J}} \in M_n$$

up to equivalence. This completes the induction step and shows that $M_{\mathcal{J}} = M_n$ holds up to equivalence.

By Theorem 7.2.6 we know that $\mathcal{K}_n$ is a base of

$$\mathrm{Th}_c(\mathbb{K}_n) \cap \mathrm{Th}(\mathbb{L}_n)$$

with background knowledge $\mathcal{S}_n = \{\, \{\, C \,\} \to \{\, D \,\} \mid C, D \in M_n, C \sqsubseteq D \,\}$. Since $M_n = M_{\mathcal{J}}$ up to equivalence, $\mathcal{K}_n$ is, up to equivalence, also a base of

$$\mathrm{Th}_c(\mathbb{K}_{\mathcal{J}}{\restriction}_{\Delta^{\mathcal{I}}}) \cap \mathrm{Th}(\mathbb{K}_{\mathcal{J}}{\restriction}_{\Delta^{\mathcal{J}} \setminus \Delta^{\mathcal{I}}})$$

with background knowledge $\mathcal{S}_n$. By Theorem 7.2.2, $\bigsqcap(\mathcal{K}_n \cup \mathcal{S}_n)$ is a base of $\mathrm{Th}_c(\mathcal{J}, \mathcal{I})$, and since $\bigsqcap(\mathcal{K}_n \cup \mathcal{S}_n)$ is element-wise equivalent to $\bigsqcap \mathcal{K}_n$, we obtain that $\bigsqcap \mathcal{K}_n$ is a base of $\mathrm{Th}_c(\mathcal{J}, \mathcal{I})$, as required. □

7.2.4. Exploring Confident GCIs with Expert Interaction

We are now prepared to devise an algorithm for model exploration by confidence. We shall achieve this by replacing in Algorithm 14 every explicit access to the interpretation $\mathcal{J}$ by suitable expert interaction. Recall that we want to do this because we now consider $\mathcal{J}$ as the background interpretation of the exploration process, which we cannot access directly.

To conduct this adaption of Algorithm 14, we observe that there are two lines in this algorithm where $\mathcal{J}$ is accessed directly, namely

i. in the computation of M_{i+1} (line 8), more precisely in the computation of the concept description $\exists r.(\bigsqcap P_i)^{\mathcal{J}\mathcal{J}}$, and

ii. in the computation of the formal context $\mathbb{L}_{i+1}$ (line 10) as the induced context of $\mathcal{J}\backslash\mathcal{I}$ and M_{i+1}.

For (i) we can argue as in Section 7.1.3, using Lemma 7.1.8: if $\mathcal{I}_\ell$ denotes the current working interpretation, which is a connected subinterpretation of the background interpretation $\mathcal{J}$, then if the expert confirms the GCI

$$\bigsqcap P_i \sqsubseteq (\bigsqcap P_i)^{\mathcal{I}_\ell\mathcal{I}_\ell},$$

then $(\bigsqcap P_i)^{\mathcal{I}_\ell\mathcal{I}_\ell} \equiv (\bigsqcap P_i)^{\mathcal{J}\mathcal{J}}$ is true.

Handling (ii) is a bit more problematic, though. A first approach would be to compute $\mathbb{L}_{i+1}$ as the induced context of $\mathcal{I}_\ell\backslash\mathcal{I}$ and M_{i+1}, where $\mathcal{I}_\ell$ is the current working interpretation. This approach, however, does not work completely: the formal context $\overline{\mathbb{L}}_{i+1}$ is used for the computation of the next candidate set P^1_{i+1}. As such, we need to ensure that all objects from $\mathbb{L}_{i+1}$ which are "relevant" for this computation are contained in $\overline{\mathbb{L}}_{i+1}$. The problem is that those objects are not necessarily counterexamples to GCIs proposed to the expert so far, and thus they may not be contained in $\mathcal{I}_\ell$. Thus, we may need to query the expert again to provide those missing objects.

Let us consider this in more detail: we observe that the computation of P^1_{i+1} in our hypothetical adaption of Algorithm 14 would *not* be correct if the lectically next intent after P_i of the current working context $\mathbb{K}_{i+1} \div \mathbb{L}_{i+1}$ in Algorithm 14 is not an intent of the working context of $\mathbb{K}_{i+1} \div \overline{\mathbb{L}}_{i+1}$. In other words

$$P^1_{i+1} = (P^1_{i+1})''_{\mathbb{K}_{i+1}\div\mathbb{L}_{i+1}}$$
$$P^1_{i+1} \neq (P^1_{i+1})''_{\mathbb{K}_{i+1}\div\overline{\mathbb{L}}_{i+1}}.$$

If $\overline{\mathcal{S}}_{i+1}$ denotes the currently known implications and $\overline{\mathcal{K}}_{i+1}$ the currently confirmed implications in our hypothetical adaption, then we know that $\overline{\mathcal{S}}_{i+1} \cup \overline{\mathcal{K}}_{i+1}$ is valid in $\overline{\mathbb{L}}_{i+1}$, and from this we obtain

$$(\mathcal{S}_{i+1} \cup \mathcal{K}_{i+1})(P^1_{i+1}) = (\mathcal{S}_{i+1} \cup \mathcal{K}_{i+1})(P^1_{i+1})''_{\mathbb{L}_{i+1}}$$
$$(\mathcal{S}_{i+1} \cup \mathcal{K}_{i+1})(P^1_{i+1}) \neq (\mathcal{S}_{i+1} \cup \mathcal{K}_{i+1})(P^1_{i+1})''_{\overline{\mathbb{L}}_{i+1}}.$$

Therefore, the implication

$$(\mathcal{S}_{i+1} \cup \mathcal{K}_{i+1})(P^1_{i+1}) \to (\mathcal{S}_{i+1} \cup \mathcal{K}_{i+1})(P^1_{i+1})''_{\overline{\mathbb{L}}_{i+1}}$$

must be rejected by the expert, because it does not hold in $\mathbb{L}_{i+1}$, and upon rejection all necessary counterexamples are added to $\overline{\mathbb{L}}_{i+1}$. Thus, if we ask all implications, or their

Algorithm 15 Model Exploration by Confidence

define `model-exploration-by-confidence`($\mathcal{I}$, c)
  $i := \ell := 0$
  $\mathcal{I}_\ell := \mathcal{I}$
  $\overline{P}_i := \overline{\mathcal{K}}_i := \varnothing$
  $\overline{M}_i := N_C \cup \{ \bot \}$
  $\overline{\mathcal{S}}_i := \{ \{ \bot \} \to \{ A \} \mid A \in N_C \}$
  choose $\leqslant_{M_i}$ as a linear order on M_i

  forever do
    ;; present new GCI to the expert
    while expert rejects $\bigsqcap \overline{P}_i \sqsubseteq (\bigsqcap \overline{P}_i)^{\mathcal{I}_\ell \mathcal{I}_\ell}$ **do**
      $\mathcal{I}_{\ell+1}$:= expert-defined extension of $\mathcal{I}_\ell$
      $\ell := \ell + 1$
    end

    $\overline{M}_{i+1} := \overline{M}_i \cup \{ \exists r.(\bigsqcap P_i)^{\mathcal{I}_\ell \mathcal{I}_\ell} \mid r \in N_R \}$ *;; union up to equivalence*
    choose $\leqslant_{M_{i+1}}$ such that (7.1) and (7.2) hold

    ;; ensure relevant counterexamples for already known GCIs
    while expert rejects $\bigsqcap \overline{P}_k \sqsubseteq \bigsqcap (\overline{P}_k^{\mathbb{K}_{\mathcal{I},\overline{M}_{i+1}},c} \cap (\overline{P}_k)^{\prime\prime_{\mathbb{K}_{\mathcal{I}_\ell \setminus \mathcal{I},\overline{M}_{i+1}}}})$ for some $k \in \{ 0, \ldots, n \}$ **do**
      $\mathcal{I}_{\ell+1}$:= expert-defined extension of $\mathcal{I}_\ell$
      $\ell := \ell + 1$
    end

    $\overline{\mathbb{K}}_{i+1}$:= `induced-context`($\mathcal{I}$, $\overline{M}_{i+1}$)
    $\overline{\mathbb{L}}_{i+1}$:= `induced-context`($\mathcal{I}_\ell \setminus \mathcal{I}$, $\overline{M}_{i+1}$)
    $\overline{\mathcal{S}}_{i+1} := \{ \{ C \} \to \{ D \} \mid C, D \in \overline{M}_{i+1}, C \sqsubseteq D \}$

    $\overline{\mathcal{K}}_{i+1} := \{ \overline{P}_k \to \overline{P}_k^{\overline{\mathbb{K}}_{i+1},c} \cap (\overline{P}_k)^{\prime\prime_{\overline{\mathbb{L}}_{i+1}}} \mid k \in \{ 0, \ldots, i \}, \overline{P}_k \neq \overline{P}_k^{\overline{\mathbb{K}}_{i+1},c} \cap (\overline{P}_k)^{\prime\prime_{\overline{\mathbb{L}}_{i+1}}} \}$

    ;; additional expert interaction
    forall $Q \geq P$ **being** $(\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1})$-closed **do**
      while expert rejects $\bigsqcap Q \sqsubseteq \bigsqcap Q^{\prime\prime_{\overline{\mathbb{L}}_{i+1}}}$ **do**
        $\mathcal{I}_{\ell+1}$:= expert-defined extension of $\mathcal{I}_\ell$
        $\ell := \ell + 1$
        $\overline{\mathbb{L}}_{i+1}$:= `induced-context`($\mathcal{I}_\ell \setminus \mathcal{I}$, $\overline{M}_{i+1}$)
      end
    end

    $\overline{P}^1_{i+1}$:= `next-closure`($\overline{M}_{i+1}$, $\leqslant_{M_{i+1}}$, $\overline{P}_i$, $\overline{\mathbb{K}}_{i+1} \div \overline{\mathbb{L}}_{i+1}$)
    $\overline{P}^2_{i+1}$:= `next-closure`($\overline{M}_{i+1}$, $\leqslant_{M_{i+1}}$, $\overline{P}_i$, $\overline{\mathcal{S}}_{i+1} \cup \overline{\mathcal{K}}_{i+1}$)

    $P_{i+1} := \min_{\leq}(P^1_{i+1}, P^2_{i+1})$.
    if $P_{i+1} =$ **nil exit**

    $i := i + 1$
  end

  return $\bigsqcap \overline{\mathcal{K}}_{i+1}$
end

corresponding GCIs, we can ensure that the computation of the set P^1_{i+1} is done as required in Algorithm 14.

Let us make this argumentation more concrete, and consider Algorithm 15 as an adaption of Algorithm 14 to provide an algorithm for model exploration by confidence. Observe that in Algorithm 15, as in Algorithm 11, counterexamples collected into the current working interpretation $\mathcal{I}_\ell$ are supposed to be connected subinterpretations of the background interpretation. Furthermore, since required by Theorem 7.2.8, we also need to ensure that $\mathcal{I}_\ell \backslash \mathcal{I}$ is a connected subinterpretation of $\mathcal{I}_\ell$, i. e. the expert is not allowed to add role-successors from elements of $\mathcal{I}_\ell \backslash \mathcal{I}$ to elements of $\mathcal{I}$ when adding counterexamples. These constraints are supposed to be satisfied in every line of the form

$$\mathcal{I}_{\ell+1} := \text{expert-defined extension of } \mathcal{I}_\ell.$$

To prove that Algorithm 15 indeed provides an algorithm for model exploration, we shall show that this algorithm computes, when applied to $\mathcal{I}$ and using an expert that represents a background interpretation $\mathcal{I}_{\mathsf{back}}$, up to equivalence the same intermediate values as Algorithm 14 when directly applied to $\mathcal{I}$ and $\mathcal{I}_{\mathsf{back}}$. Then, since Algorithm 14 terminates, Algorithm 15 will also terminate and returns a base of $\mathrm{Th}_c(\mathcal{I}_{\mathsf{back}}, \mathcal{I})$.

7.2.10 Theorem *Let $\mathcal{I}_{\mathsf{back}}$ be a finite interpretation over N_C and N_R, and let $\mathcal{I}$ be a connected subinterpretation of $\mathcal{I}_{\mathsf{back}}$ such that $\mathcal{I}_{\mathsf{back}} \backslash \mathcal{I}$ is also a connected subinterpretation of $\mathcal{I}_{\mathsf{back}}$. Let $c \in [0, 1]$.*

Then Algorithm 15 applied to $\mathcal{I}$, c and using an expert that represents $\mathcal{I}_{\mathsf{back}} \backslash \mathcal{I}$ terminates. If n is the number of iterations of this run, then $\bigcap \overline{\mathcal{K}}_n$ is a base of $\mathrm{Th}_c(\mathcal{I}_{\mathsf{back}}, \mathcal{I}) = \mathrm{Th}_c(\mathcal{I}) \cap \mathrm{Th}(\mathcal{I}_{\mathsf{back}} \backslash \mathcal{I})$.

Proof We show that Algorithm 15 with the input $\mathcal{I}_{\mathsf{back}}$, $\mathcal{I}$ and c has the same output as Algorithm 14. Then the claim follows from Theorem 7.2.8.

To this end, we shall show that for all $i \in \{0, \ldots, n\}$ it is true that $\overline{P}_i = P_i, \overline{M}_i = M_i, \overline{\mathcal{K}}_i = \mathcal{K}$ and $\overline{\mathcal{S}}_i = \mathcal{S}_i$ is true up to equivalence. In other words, we shall show that every element of $\overline{P}_i$ is equivalent to some element in P_i, and vice versa; likewise for $\overline{M}_i$ and M_i. Furthermore, we shall show that for each implication $(\overline{A} \to \overline{B}) \in \overline{\mathcal{K}}_i$ there exists an implication $(A \to B) \in \mathcal{K}_i$ such that $\overline{A} = A$ and $\overline{B} = B$ is true up to equivalence, and vice versa; likewise for $\overline{\mathcal{S}}_i$ and $\mathcal{S}_i$.

We shall show these claims by induction over i.

In the following argumentation, we shall not mention the linear orders we choose on the sets $\overline{M}_{i+1}$ and M_{i+1} explicitly. However, we shall choose the linear orders $\leqslant_{\overline{M}_{i+1}}$ and $\leqslant_{M_{i+1}}$ on $\overline{M}_{i+1}$ and M_{i+1} such that

$$\overline{C} \leqslant_{\overline{M}_{i+1}} \overline{D} \iff C \leqslant_{M_{i+1}} D$$

for $\overline{C}, \overline{D} \in \overline{M}_{i+1}, C, D \in M_{i+1}$ and $\overline{C} \equiv C, \overline{D} \equiv D$.

Base Case: For $i = 0$, it is true that $\overline{P}_i = \varnothing = P_i$, $\overline{M}_i = N_C \cup \{ \bot \} = M_i$, $\overline{\mathcal{K}}_i = \varnothing = \mathcal{K}_i$ and $\overline{\mathcal{S}}_i = \{ \{ \bot \} \to \{ A \} \mid A \in N_C \} = \mathcal{S}_i$. Thus, the claim holds for $i = 0$.

Step Case: Let us assume that $0 < i < n$ and that the claim holds for all $m \leqslant i$.

Denote with $\mathcal{I}_l$ the current working interpretation when the algorithm has reached line 15. The algorithm can only reach this line if the expert has confirmed $\bigsqcap \overline{P}_i \sqsubseteq (\bigsqcap \overline{P}_i)^{\mathcal{I}_l \mathcal{I}_l}$. Since $\mathcal{I}_l$ is a connected subinterpretation of $\mathcal{I}_{\mathsf{back}}$, Lemma 7.1.8 implies that $(\bigsqcap \overline{P}_i)^{\mathcal{I}_l \mathcal{I}_l} = (\bigsqcap \overline{P}_i)^{\mathcal{I}_{\mathsf{back}} \mathcal{I}_{\mathsf{back}}}$. Since $\overline{M}_i = M_i$ holds up to equivalence by induction hypothesis, it is true that $\overline{M}_{i+1} = M_{i+1}$ up to equivalence. This also implies $\overline{\mathcal{S}}_i = \mathcal{S}_i$ up to equivalence, since the definition of these sets only depends on $\overline{M}_{i+1}$ and M_i, respectively.

To show $\overline{\mathcal{K}}_i = \mathcal{K}$, it is sufficient to verify that

$$\overline{P}_k^{\overline{\mathbb{K}}_{i+1}, \mathcal{C}} \cap (\overline{P}_k)''_{\overline{\mathbb{L}}_{i+1}} = P_k^{\mathbb{K}_{i+1}, \mathcal{C}} \cap (P_k)''_{\mathbb{L}_{i+1}} \tag{7.9}$$

is true up to equivalence for all $k \in \{ 0, \ldots, i \}$. To this end, we first observe that $\overline{\mathbb{K}}_{i+1}$ is the induced context of $\mathcal{I}$ and $\overline{M}_{i+1}$, and $\mathbb{K}_{i+1}$ is the induced context of $\mathcal{I}$ and M_{i+1}. In particular, since $\overline{M}_{i+1} = M_{i+1}$ and $\overline{P}_k = P_k$ up to equivalence for all $0 \leqslant k \leqslant i$, we obtain

$$\overline{P}_k^{\overline{\mathbb{K}}_{i+1}, \mathcal{C}} = P_k^{\mathbb{K}_{i+1}, \mathcal{C}}$$

up to equivalence for all $k \in \{ 0, \ldots, i \}$.

Let $\mathcal{I}_m$ be the current working interpretation in iteration i when line 24 is reached. Recall that $\overline{\mathbb{L}}_{i+1}$ is the induced context of $\mathcal{I}_m \backslash \mathcal{I}$ and $\overline{M}_{i+1}$, and that $\mathbb{L}_{i+1}$ is the induced context of $\mathcal{I}_{\mathsf{back}} \backslash \mathcal{I}$ and M_{i+1}. Since $\mathcal{I}_m \backslash \mathcal{I}$ is a connected subinterpretation of $\mathcal{I}_{\mathsf{back}} \backslash \mathcal{I}$, we can consider $\overline{\mathbb{L}}_{i+1}$ as a subcontext of $\mathbb{L}_{i+1}$, and thus obtain

$$(\overline{P}_k)''_{\overline{\mathbb{L}}_{i+1}} \supseteq (P_k)''_{\mathbb{L}_{i+1}}$$

up to equivalence for all $k \in \{ 0, \ldots, i \}$.

Assume now by contradiction that

$$\overline{P}_k^{\overline{\mathbb{K}}_{i+1}, \mathcal{C}} \cap (\overline{P}_k)''_{\overline{\mathbb{L}}_{i+1}} \neq P_k^{\mathbb{K}_{i+1}, \mathcal{C}} \cap (P_k)''_{\mathbb{L}_{i+1}}$$

is true for some $k \in \{ 0, \ldots, i \}$. By the above considerations, this means that there exists a concept description $\overline{C} \in \overline{P}_k^{\overline{\mathbb{K}}_{i+1}, \mathcal{C}} \cap (\overline{P}_k)''_{\overline{\mathbb{L}}_{i+1}}$ that is not equivalent to any element in $P_k^{\mathbb{K}_{i+1}, \mathcal{C}} \cap (P_k)''_{\mathbb{L}_{i+1}}$. Then

i. the expert has confirmed the implication

$$\bigsqcap \overline{P}_k \sqsubseteq \bigsqcap \overline{P}_k^{\overline{\mathbb{K}}_{i+1}, \mathcal{C}} \cap (\overline{P}_k)''_{\overline{\mathbb{L}}_{i+1}}$$

since it was proposed to her in line 19. In particular, the GCI $\bigsqcap \overline{P}_k \sqsubseteq \overline{C}$ is valid in $\mathcal{I}_{\mathsf{back}} \backslash \mathcal{I}$.

ii. The GCI $\bigcap P_k \sqsubseteq \overline{C}$ is not confirmed by the expert. To see this, observe that since $\overline{C} \in M_{i+1}$, there exists $C \in M_{i+1}$ such that $\overline{C} \equiv C$. Then if $\bigcap P_k \sqsubseteq \overline{C}$ were confirmed by the expert, it would be true that $C \in (P_k)''_{\mathbb{L}_{i+1}}$. Furthermore, it is true that $\mathrm{conf}_{\overline{\mathbb{K}}_{i+1}}(\overline{P}_k \to \{\overline{C}\}) = \mathrm{conf}_{\mathbb{K}_{i+1}}(P_k \to \{C\})$ because of $\overline{P}_k = P_k$, $\overline{M}_{i+1} = M_{i+1}$ up to equivalence, and $\overline{C} \equiv C$. Since $\overline{C} \in \overline{P}_k^{\overline{\mathbb{K}}_{i+1},c}$, it is true that $\mathrm{conf}_{\overline{\mathbb{K}}_{i+1}}(\overline{P}_k \to \{\overline{C}\}) \geqslant c$, and thus $\mathrm{conf}_{\mathbb{K}_{i+1}}(P_k \to \{C\}) \geqslant c$, and hence $C \in P_k^{\mathbb{K}_{i+1},c}$. Thus, we obtain

$$\overline{C} \equiv C \quad \text{and} \quad C \in P_k^{\mathbb{K}_{i+1},c} \cap (P_k)''_{\mathbb{L}_{i+1}},$$

contradicting our choice of $\overline{C}$. Therefore, $\bigcap P_k \sqsubseteq \overline{C}$ is not confirmed by the expert, and in particular is not valid in $\mathcal{I}_{\mathsf{back}} \backslash \mathcal{I}$.

However, since $\overline{P}_k = P_k$ up to equivalence, the fact that $\bigcap \overline{P}_k \sqsubseteq \overline{C}$ is valid in $\mathcal{I}_{\mathsf{back}} \backslash \mathcal{I}$ but $\bigcap P_k \sqsubseteq \overline{C}$ is not, yields the desired contradiction. Therefore, we have established the validity of Equation (7.9), and thus can infer that $\overline{\mathcal{K}}_k = \mathcal{K}_k$ is true up to equivalence.

It remains to be shown that

$$\overline{P}_{i+1} = P_{i+1} \tag{7.10}$$

is true up to equivalence. To this end, we observe that $\overline{P}^2_{i+1} = P^2_{i+1}$ is true up to equivalence, since $\overline{P}_i = P_i$ and $\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1} = \mathcal{K}_{i+1} \cup \mathcal{S}_{i+1}$ up to equivalence, and the linear orders on $\overline{M}_{i+1}$ and M_i are chosen suitably. Thus, to show Equation (7.10) it suffices to verify that

$$\overline{P}^1_{i+1} = P^1_{i+1}. \tag{7.11}$$

For this recall that the formal contexts $\overline{\mathbb{K}}_{i+1}$ and $\mathbb{K}_{i+1}$ can be considered the same, because $\overline{M}_{i+1} = M_{i+1}$ up to equivalence. Also recall that we can consider $\overline{\mathbb{L}}_{i+1}$ as a subcontext of $\mathbb{L}_{i+1}$, again up to equivalence. Therefore, we obtain $\overline{P}^1_{i+1} \geq P^1_{i+1}$ up to equivalence.

Suppose by contradiction that $\overline{P}^1_{i+1} \not\geq P^1_{i+1}$. Then P^1_{i+1}, viewed as a subset of $\overline{M}_{i+1}$, is not an intent of $\overline{\mathbb{K}}_{i+1} \div \overline{\mathbb{L}}_{i+1}$, as otherwise $\overline{P}^1_{i+1} \preceq P^1_{i+1}$. Since P^1_{i+1} is an intent of $\mathbb{K}_{i+1} \div \mathbb{L}_{i+1}$, we can thus infer that

$$(P^1_{i+1})''_{\overline{\mathbb{L}}_{i+1}} \backslash (P^1_{i+1})''_{\mathbb{L}_{i+1}} \neq \varnothing.$$

Let $D \in (P^1_{i+1})''_{\overline{\mathbb{L}}_{i+1}} \backslash (P^1_{i+1})''_{\mathbb{L}_{i+1}}$. Since $\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1}$ is sound for $\mathbb{L}_{i+1}$ up to equivalence, we obtain that

$$((\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1})(P^1_{i+1}))''_{\mathbb{L}_{i+1}} = (P^1_{i+1})''_{\mathbb{L}_{i+1}}$$

and thus

$$D \notin ((\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1})(P^1_{i+1}))''_{\mathbb{L}_{i+1}}.$$

Therefore, the corresponding implication

$$(\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1})(P^1_{i+1}) \to ((\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1})(P^1_{i+1}))''_{\mathbb{L}_{i+1}}$$

does not hold in $\mathbb{L}_{i+1}$, because

$$((\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1})(P^1_{i+1}))''_{\mathbb{L}_{i+1}} \not\subseteq ((\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1})(P^1_{i+1}))''_{\mathbb{L}_{i+1}}.$$

Therefore, the corresponding GCI

$$\bigsqcap(\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1})(P^1_{i+1}) \sqsubseteq \bigsqcap((\overline{\mathcal{K}}_{i+1} \cup \overline{\mathcal{S}}_{i+1})(P^1_{i+1}))''_{\mathbb{L}_{i+1}} \tag{7.12}$$

will be rejected by the expert, since $\mathbb{L}_{i+1}$ is the induced context of $\mathcal{I}_{\mathsf{back}} \backslash \mathcal{I}$.

However, when computing P^1_{i+1}, we have passed the lines 31 up to 37, and the expert has confirmed the GCI given in Equation (7.12). Therefore, our initial assumption that $\overline{P}^1_{i+1} \gneq P^1_{i+1}$ is not true, and thus we obtain $\overline{P}^1_{i+1} = P^1_{i+1}$ up to equivalence. This finishes the proof. □

CHAPTER 8

Conclusions and Outlook

This work is concerned with axiomatizing all GCIs expressible in the description logic $\mathcal{EL}^\bot$ that enjoy a certain minimal confidence in a given finite interpretation. The motivation for this research stemmed from the observation that the results from Baader and Distel [41] on finding finite bases of finite interpretations are very sensitive towards sporadic errors in the given interpretation, and thus may not perform well on real-world data, which can be assumed to always contain errors.

The way errors affect the computation of bases of finite interpretations has been assessed in detail in Section 5.1, where we have argued how *linked data* can be seen as a variant of finite interpretations, and where we have computed finite bases of the finite interpretation $\mathcal{I}_{\text{DBpedia}}$ that has been obtained from the DBpedia data set. Computing finite bases of the interpretation $\mathcal{I}_{\text{DBpedia}}$ turned out to be not very insightful, mainly because the concept names contained in $\mathcal{I}_{\text{DBpedia}}$ consisted mostly of job descriptions, and it cannot be expected that the job of a parent affects the jobs of their children, or vice versa, in a way expressible as a GCI. Furthermore, the interpretation $\mathcal{I}_{\text{DBpedia}}$ turned out to be erroneous to the extent that it contained things that are usually not associated with the child role, like places, books or bands.

Despite the deficits of $\mathcal{I}_{\text{DBpedia}}$, we have argued that one still can expect certain GCIs to be valid in $\mathcal{I}_{\text{DBpedia}}$, most notably

$$\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}.$$

However, this GCI turned out to be not valid in $\mathcal{I}_{\text{DBpedia}}$. The reason for this was that among the 2551 elements to which this GCI is applicable, four of them were erroneously not an instance of Person. In other words, the GCI $\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}$ will not be collected because it is erroneously invalidated by a comparably small number of counterexamples.

This observation motivated us to extend the results on axiomatizing valid GCIs into the direction of axiomatizing GCIs with *high confidence* in the given data. This extension has been discussed in Section 5.2, where we had adapted results obtained by Luxenburger [69] on *partial implications* in finite contexts to the setting of GCIs in finite interpretations $\mathcal{I}$.

From this adaption we have obtained finite confident bases of finite interpretations (Theorem 5.2.11, Theorem 5.2.13), and a way to compute finite confident bases of GCIs from finite confident bases of implications (Theorem 5.2.22). We have also seen how we can make use of the canonical base of the induced context $\mathbb{K}_{\mathcal{I}}$ of $\mathcal{I}$ to complete bases of $\mathrm{Th}_c(\mathcal{I}) \setminus \mathrm{Th}(\mathcal{I})$ to confident bases of $\mathrm{Th}_c(\mathcal{I})$. Finally, in Section 5.2.6 we have discussed a way to obtain finite confident $\mathcal{EL}^{\bot}$ bases from finite confident $\mathcal{EL}^{\bot}_{\mathrm{gfp}}$ bases, using the technique of *unravelling* $\mathcal{EL}^{\bot}_{\mathrm{gfp}}$ concept descriptions.

The results we had obtained about computing finite confident bases of $\mathcal{I}$ were then in return applied to our example interpretation $\mathcal{I}_{\mathrm{DBpedia}}$, to examine how the approach of computing finite confident bases performs on this data set. As expected, we could recover the GCI $\exists \mathsf{child}.\top \sqsubseteq \mathsf{Person}$, as well as another GCI. In particular, we have observed that for relatively large values of the confidence threshold c, the number of additional GCIs to be examined by the expert is relatively small. Thus, one could argue that the approach of considering GCIs with high confidence does not add much overhead for practical applications, but promises to deliver more robust results.

However, the approach of considering GCIs with high confidence is still a heuristic approach, and the GCIs thus obtained need to be validated by an external source of information, like a human expert. Indeed, the same is true for valid GCIs of the input interpretation $\mathcal{I}$, as it may be the case that $\mathcal{I}$ misses some crucial counterexamples for the domain of interest. For this purpose, *model exploration* has been developed to interactively consult an expert during the computation of a finite base of $\mathcal{I}$ to provide missing counterexamples as needed. We have argued that providing such an exploration algorithm for the computation of finite confident bases may also be helpful for practical applications.

To this end, we have first discussed in Chapter 6 an approach to obtain an exploration algorithm for *implications with high confidence* in some given formal context $\mathbb{K}$, called *exploration by confidence*. For this, we have introduced in Section 6.1 an abstraction of the classical attribute exploration algorithm as an algorithm to explore the set $\mathrm{Th}(\mathbb{K})$ of implications. Then in Section 6.2.1, we have argued how this abstraction can be transferred to the set $\mathrm{Th}_c(\mathbb{K})$, automatically yielding an algorithm that explores the set $\mathrm{Th}_c(\mathbb{K})$. This algorithm has the practical disadvantages that it requires the computation of closures under $\mathrm{Th}_c(\mathbb{K})$, which may not be feasible in certain applications, and that it is only approximative. To remedy this, we have introduced in Section 6.2.2 an exploration algorithm that avoids these shortcomings, but may ask more questions to the expert.

The purpose of the then following Chapter 7 was to extend the algorithm for exploration by confidence to provide an algorithm for *model exploration by confidence*. Such an algorithm was obtained in Section 7.2.4. As a byproduct of the argumentation used, we also obtained more practical algorithms for computing confident bases of formal contexts (Section 7.2.2) and of finite interpretations (Section 7.2.3). Those algorithms compute the set $M_{\mathcal{I}}$ incrementally, and not completely as a first step, thus avoiding the delay of the latter when computing implications and GCIs, respectively.

The work presented so far is not complete, and a lot remains to be done to turn the

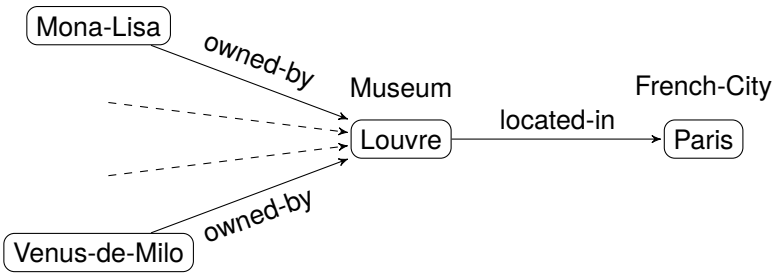


Figure 8.1.: Support can be counter-intuitive

results of this thesis into practically relevant techniques. As a first step one could extend the results of this thesis to also incorporate the notion of *support* of GCIs, an idea which is again borrowed from standard data mining approaches [1]. More precisely, one could define for a GCI $C \sqsubseteq D$ its support $\mathrm{supp}_{\mathcal{I}}(C \sqsubseteq D)$ in $\mathcal{I}$ to be

$$\mathrm{supp}_{\mathcal{I}}(C \sqsubseteq D) := \frac{|(C \sqcap D)^{\mathcal{I}}|}{|\Delta^{\mathcal{I}}|}.$$

Then one can consider only those GCIs whose support is above a given threshold $s \in [0, 1]$. The idea behind this approach is that if the support of $C \sqsubseteq D$ is not high enough in $\mathcal{I}$, then the data set does not contain enough information about this GCI to decide whether $C \sqsubseteq D$ is true in the domain of discourse or not. In latter case, the GCI $C \sqsubseteq D$ should be ignored.

An adaption of the results presented in this thesis to include the support measure should not be very difficult, and indeed results from formal concept analysis could again be used here [96, 97]. However, one could also step back and ask whether the notions of support and confidence are really suitable for our description logic setting: it has been argued in [33] that the support (and also the confidence) of a GCI as defined above may be counter-intuitive, at least as measures of trustworthiness. To see this, let us consider the example as given in [33], which consists of an interpretation that contains an element for the Louvre in Paris, all pieces of art that are kept there, and an element to represent the city of Paris. The complete example interpretation is sketched in Figure 8.1.

In this example interpretation, the GCI

$$\mathsf{Museum} \sqsubseteq \exists \mathsf{located\text{-}in}.\mathsf{French\text{-}City}$$

has very low support, because there is only one museum in the data set. Therefore, one could argue that this GCI is not very trustworthy. Based on this, one could argue that the GCI

$$\exists \mathsf{owned\text{-}by}.\mathsf{Museum} \sqsubseteq \exists \mathsf{owned\text{-}by}.\exists \mathsf{located\text{-}in}.\mathsf{French\text{-}City}$$

is not trustworthy either. However, this GCI has very high support in the example interpretation, because there are thousands of exhibits in the Louvre. Therefore, the support

measure would assign this GCI a very high trustworthiness, although we would intuitively assign it a rather low one. Finding a more intuitive approach to measure trustworthiness and relevance of GCIs in a given data set remains an open problem.

But even if one accepts the way we have defined the confidence of an GCI, more work remains to be done to make our approach more practically relevant. For example, in Theorem 5.2.22 we have shown a way how finite confident bases of finite interpretations can be gained from bases of implications with high confidence in the corresponding induced formal context. What we did not talk about was a way to compute such bases of implications with high confidence. To compute these, it may prove fruitful to exploit the close connection of formal concept analysis to data mining [105], and try to adapt known algorithms from data mining for computing association rules to the setting of computing bases of implications with high confidence. In this way, one could hope to obtain much faster implementations as those we have used for our experiments [33], and thus increase the practicability of our approach to construct knowledge bases from large amounts of data.

Another usability issue concerns the way we have constructed our algorithm for model exploration by confidence, as presented in Section 7.2.4. There, the number of GCIs asked to the expert may be quite high, probably much higher then in the model exploration algorithm as proposed by Baader and Distel. Moreover, if the GCIs are presented to a human expert, extra care is needed. More precisely, the GCIs

$$\bigsqcap P_i \sqsubseteq (\bigsqcap P_i)^{\mathcal{I}_\ell \mathcal{I}_\ell}$$

we ask to the expert may contain proper $\mathcal{EL}^{\bot}_{\text{gfp}}$ concept descriptions, and the size of the premises P_i may be too large, both being issues that may inhibit human comprehension.

The former problem can be attacked by considering only those GCIs where the role-depth is not larger than an a-priori chosen maximal role-depth $k \in \mathbb{N}$. More precisely, we can define

$$\mathrm{Th}^k_c(\mathcal{I}) := \{\, (C \sqsubseteq D) \in \mathrm{Th}_c(\mathcal{I}) \mid d(C), d(D) \leqslant k \,\},$$

and can then try to find small bases $\mathcal{B} \subseteq \mathrm{Th}^k_c(\mathcal{I})$ of $\mathrm{Th}^k_c(\mathcal{I})$ (note that now $\mathrm{Th}^k_c(\mathcal{I})$ is finite, up to equivalence). In the case $c = 1$, where we only consider valid GCIs of $\mathcal{I}$, all results obtained by Baader and Distel can be carried over, as sketched in [39]. Indeed, limiting the role-depth of the concept descriptions involved eliminates the need for the description logic $\mathcal{EL}^{\bot}_{\text{gfp}}$, and thus simplifies the underlying theory considerably. Transferring these results to the setting of GCIs with high confidence seems practically relevant.

To address the fact that P_i may be too large to be understandable for human experts, one could restrict oneself to only find bases of GCIs $C \sqsubseteq D$ where C has at most ℓ conjuncts, for some a-priori chosen value $\ell \in \mathbb{N}$. To solve the problem of finding bases for this set of GCIs, one could first try to solve the simpler problem of finding a base of the set of valid

implications in a formal context $\mathbb{K} = (G, M, I)$ where the premise has size of at most ℓ. Note that such a base would be

$$\{ A \to A'' \mid A \subseteq M, |A| \leqslant \ell \}.$$

However, this base has size $\binom{|M|}{\ell}$, which, although polynomial in $|M|$, may be too large for practical applications, and thus finding a smaller base is necessary. It would also be desirable to have an exploration algorithm that only asks implications where the premise does not have more than ℓ elements. As soon as those problems are solved for implications, one should try to transfer them to the setting of GCIs, probably again with an a-priori chosen maximal role-depth for the concept descriptions to be considered.

Chapter A

Bibliography

[1] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. "Mining Association Rules between Sets of Items in Large Databases". In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 1993, pp. 207–216 (cit. on pp. 2, 84, 91, 191).

[2] Dana Angluin. "Queries and Concept Learning". In: *Machine Learning* 2.4 (1987), pp. 319–342 (cit. on p. 11).

[3] Dana Angluin, Michael Frazier, and Leonard Pitt. "Learning Conjunctions of Horn Clauses". In: *Machine Learning* 9 (1992), pp. 147–164 (cit. on p. 11).

[4] Marta Arias and José L. Balcázar. "Construction and learnability of canonical Horn formulas". In: *Machine Learning* 85.3 (2011), pp. 273–297 (cit. on p. 11).

[5] Michael Ashburner et al. "Gene ontology: tool for the unification of biology". In: *Nature Genetics* 25 (May 2000), pp. 25–29 (cit. on p. 3).

[6] Franz Baader. "Computing a Minimal Representation of the Subsumption Lattice of all Conjunctions of Concepts Defined in a Terminology". In: *Proceedings of the International Symposium on Knowledge Retrieval, Use, and Storage for Efficiency*. 1995, pp. 168–178 (cit. on p. 9).

[7] Franz Baader. "Computing the Least Common Subsumer in the Description Logic $\mathcal{EL}$ w.r.t. Terminological Cycles with Descriptive Semantics". In: *Proceedings of the 11th International Conference on Conceptual Structures*. (Dresden, Germany). Ed. by Aldo de Moor, Wilfried Lex, and Bernhard Ganter. Vol. 2746. Lecture Notes in Computer Science. Springer, 2003, pp. 117–130 (cit. on p. 61).

[8] Franz Baader. "Least Common Subsumers and Most Specific Concepts in a Description Logic with Existential Restrictions and Terminological Cycles". In: *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*. (Acapulco, Mexico). Ed. by Georg Gottlob and Toby Walsh. Morgan Kaufmann, Aug. 2003, pp. 319–324 (cit. on p. 61).

[9] Franz Baader. "Terminological Cycles in a Description Logic with Existential Restrictions". In: *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*. (Acapulco, Mexico). Ed. by Georg Gottlob and Toby Walsh. Morgan Kaufmann, Aug. 2003, pp. 325–330 (cit. on pp. 3, 56, 58, 63, 70).

[10] Franz Baader, Sebastian Brandt, and Carsten Lutz. "Pushing the $\mathcal{EL}$ Envelope". In: *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*. (Edinburgh, Scotland, United Kingdom). Ed. by Leslie Pack Kaelbling and Alessandro Saffiotti. Morgan Kaufmann, July 2005, pp. 364–369 (cit. on pp. 3, 56).

[11] Franz Baader, Sebastian Brandt, and Carsten Lutz. "Pushing the $\mathcal{EL}$ Envelope Further". In: *Proceedings of the OWLED 2008 DC Workshop on OWL: Experiences and Directions*. Ed. by Kendall Clark and Peter F. Patel-Schneider. 2008 (cit. on p. 3).

[12] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, eds. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003 (cit. on pp. 1, 3, 49, 56).

[13] Franz Baader and Felix Distel. "A Finite Basis for the Set of $\mathcal{EL}$-Implications Holding in a Finite Model". In: *Proceedings of the 6th International Conference on Formal Concept Analysis*. (Montreal, Canada). Ed. by Raoul Medina and Sergei A. Obiedkov. Vol. 4933. Lecture Notes in Computer Science. Springer, Feb. 2008, pp. 46–61 (cit. on p. 8).

[14] Franz Baader and Felix Distel. "Exploring Finite Models in the Description Logic $\mathcal{EL}_{\text{gfp}}$". In: *Proceedings of the 7th International Conference on Formal Concept Analysis*. (Darmstadt, Germany). Ed. by Sébastien Ferré and Sebastian Rudolph. Vol. 5548. Lecture Notes in Computer Science. Springer, May 2009, pp. 146–161 (cit. on p. 8).

[15] Franz Baader, Bernhard Ganter, Barış Sertkaya, and Ulrike Sattler. "Completing Description Logic Knowledge Bases Using Formal Concept Analysis". In: *Proceedings of the 20th International Joint Conference on Artificial Intelligence*. (Hyderabad, India). Ed. by Manuela M. Veloso. Jan. 2007, pp. 230–235 (cit. on pp. 8, 9).

[16] Franz Baader, Ralf Küsters, and Ralf Molitor. "Computing Least Common Subsumers in Description Logics with Existential Restrictions". In: *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*. (Stockholm, Sweden). Ed. by Thomas Dean. Morgan Kaufmann, July 1999, pp. 96–103 (cit. on p. 57).

[17] Franz Baader and Carsten Lutz. "Description Logic". In: *Handbook of Modal Logic*. Ed. by Patrick Blackburn, Johan van Benthem, and Frank Wolter. Elsevier Science, 2006, pp. 757–820 (cit. on p. 52).

[18] Franz Baader and Ulrike Sattler. "An Overview of Tableau Algorithms for Description Logics". In: *Studia Logica* 69.1 (2001), pp. 5–40 (cit. on p. 3).

[19] Mikhail A. Babin and Sergei O. Kuznetsov. "Computing premises of a minimal cover of functional dependencies is intractable". In: *Discrete Applied Mathematics* 161.6 (2013), pp. 742–749 (cit. on pp. 7, 43).

[20] Marc Barbut and Bernard Monjardet. *Ordre et Classification – Algèbre et Combinatoire*. Paris: Hachette, 1970 (cit. on p. 5).

[21] Radim Bělohlávek and Vilem Vychodil. "What is a fuzzy concept lattice?" In: *Proceedings of the International Workshop on Concept Lattices and their Applications*. Ed. by Radim Bělohlávek and Vaclav Snasel. Vol. 162. CEUR Workshop Proceedings. CEUR-WS.org, 2005, pp. 34–45 (cit. on p. 11).

[22] Tim Berners-Lee, James Hendler, and Ora Lassila. "The Semantic Web". In: *Scientific American Magazine* (2001) (cit. on p. 83).

[23] Garrett Birkhoff. *Lattice Theory*. 3rd ed. Vol. 25. AMS Colloquium Publications. American Mathematical Society, 1967 (cit. on pp. 5, 18).

[24] Christian Bizer, Tom Heath, and Tim Berners-Lee. "Linked Data - The Story So Far". In: *International Journal on Semantic Web and Information Systems* 5.3 (Mar. 2009). Ed. by Tom Heath, Martin Hepp, and Christian Bizer, pp. 1–22 (cit. on p. 83).

[25] Christian Bizer et al. "DBpedia - A Crystallization Point of the Web of Data". In: *Web Semantics: Science, Services and Agents on the World Wide Web* 7.3 (2009), pp. 154–165 (cit. on pp. 11, 83, 85).

[26] Patrick Blackburn, Johan van Benthem, and Frank Wolter, eds. *Handbook of Modal Logic*. Elsevier Science, 2006 (cit. on p. 52).

[27] Fernando Bobillo and Umberto Straccia. "Fuzzy description logics with general t-norms and datatypes". In: *Fuzzy Sets and Systems* 160.23 (2009), pp. 3382–3402 (cit. on p. 11).

[28] Daniel Borchmann. *Axiomatizing Confident $\mathcal{EL}^{\perp}_{gfp}$-GCIs of Finite Interpretations*. Tech. rep. MATH-AL-08-2012. Chair of Algebraic Structure Theory, Institute of Algebra, Technische Universität Dresden, Sept. 2012 (cit. on pp. 83, 107).

[29] Daniel Borchmann. "Axiomatizing Confident $\mathcal{EL}^{\perp}_{\text{gfp}}$-General Concept Inclusions in the Presence of Untrusted Individuals". In: *Informal Proceedings of the 26th International Workshop on Description Logics*. (Ulm, Germany). Ed. by Thomas Eiter, Birte Glimm, Yevgeny Kazakov, and Markus Krötzsch. Vol. 1014. CEUR Workshop Proceedings. CEUR-WS.org, July 2013, pp. 65–79 (cit. on p. 167).

[30] Daniel Borchmann. *Exploration by Confidence*. LTCS-Report 13-04. Chair of Automata Theory, Institute of Theoretical Computer Science, Technische Universität Dresden, 2013 (cit. on pp. 125, 132).

[31] Daniel Borchmann. *Model Exploration by Confidence with Completely Specified Counterexamples*. LTCS-Report 13-11. Chair of Automata Theory, Institute of Theoretical Computer Science, Technische Universität Dresden, 2013 (cit. on p. 156).

[32] Daniel Borchmann. *On Confident GCIs of Finite Interpretations*. LTCS-Report 12-06. Chair of Automata Theory, Institute of Theoretical Computer Science, Technische Universität Dresden, 2012 (cit. on pp. 102, 116).

[33] Daniel Borchmann and Felix Distel. "Mining of $\mathcal{EL}$-GCIs". In: *Proceedings of the 11th International Conference on Data Mining, Workshops*. (Vancouver, British Columbia, Canada). Ed. by Myra Spiliopoulou et al. IEEE, Dec. 2011, pp. 1083–1090 (cit. on pp. 8, 43, 83, 191, 192).

[34] Ronald J. Brachman and Hector J. Levesque. "The Tractability of Subsumption in Frame-Based Description Languages". In: *Proceedings of the National Conference on Artificial Intelligence*. (Austin, Texas, USA). Ed. by Ronald J. Brachman. AAAI Press, Aug. 1984, pp. 34–37 (cit. on p. 3).

[35] Ronald J. Brachman and James G. Schmolze. "An Overview of the KL-ONE Knowledge Representation System". In: *Cognitive Science* 9.2 (1985), pp. 171–216 (cit. on p. 3).

[36] Sebastian Brandt. "Polynomial Time Reasoning in a Description Logic with Existential Restrictions, GCI Axioms, and - What Else?" In: *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, including Prestigious Applicants of Intelligent Systems*. (Valencia, Spain). Ed. by Ramon López de Mántaras and Lorenza Saitta. IOS Press, Aug. 2004, pp. 298–302 (cit. on pp. 3, 56).

[37] Peter Burmeister and Richard Holzer. "Treating Incomplete Knowledge in Formal Concept Analysis". In: *Formal Concept Analysis, Foundations and Applications*. Ed. by Bernhard Ganter, Gerd Stumme, and Rudolf Wille. Vol. 3626. Lecture Notes in Computer Science. Springer, 2005, pp. 114–126 (cit. on p. 8).

[38] Brian A. Davey and Hilary A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2002 (cit. on p. 18).

[39] Felix Distel. *Formal Concept Analysis and Logics*. Lecture Notes. 2012 (cit. on p. 192).

[40] Felix Distel. "Hardness of Enumerating Pseudo-intents in the Lectic Order". In: *Proceedings of the 8th Interational Conference of Formal Concept Analysis*. (Agadir, Morocco). Ed. by Léonard Kwuida and Barış Sertkaya. Vol. 5986. Lecture Notes in Computer Science. Springer, Mar. 2010, pp. 124–137 (cit. on pp. 7, 43).

[41] Felix Distel. "Learning Description Logic Knowledge Bases from Data Using Methods from Formal Concept Analysis". PhD thesis. Technische Universität Dresden, 2011 (cit. on pp. 2, 5, 8, 34, 47, 49, 63, 64, 67, 70, 71, 73–75, 78–82, 108, 112, 123, 155, 156, 158, 159, 161–166, 174, 179, 180, 189).

[42] William F. Dowling and Jean H. Gallier. “Linear-Time Algorithms for Testing the Satisfiability of Propositional Horn Formulae”. In: *Journal of Logic Programming* 1.3 (1984), pp. 267–284 (cit. on p. 31).

[43] Dieter Fensel, James A. Hendler, Henry Lieberman, and Wolfgang Wahlster, eds. *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential [outcome of a Dagstuhl seminar]*. MIT Press, 2003 (cit. on p. 83).

[44] Sébastien Ferré and Olivier Ridoux. “A Logical Generalization of Formal Concept Analysis”. In: *Conceptual Structures: Logical, Linguistic, and Computational Issues; Proceedings of the 8th International Conference on Conceptual Structures*. (Darmstadt, Germany). Ed. by Bernhard Ganter and Guy W. Mineau. Vol. 1867. Lecture Notes in Computer Science. Springer, Aug. 2000, pp. 371–384 (cit. on p. 5).

[45] Bernhard Ganter. “Attribute Exploration with Background Knowledge”. In: *Theoretical Computer Science* 217.2 (1999), pp. 215–233 (cit. on pp. 8, 47).

[46] Bernhard Ganter. “Two Basic Algorithms in Concept Analysis”. In: *Proceedings of the 8th Interational Conference of Formal Concept Analysis*. (Agadir, Morocco). Ed. by Léonard Kwuida and Barış Sertkaya. Vol. 5986. Lecture Notes in Computer Science. Springer, Mar. 2010, pp. 312–340 (cit. on pp. 7, 38, 44).

[47] Bernhard Ganter, Sergei Obiedkov, Sebastian Rudolph, and Gerd Stumme. “Conceptual Exploration”. In preparation (cit. on pp. 7, 126).

[48] Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, 1999 (cit. on pp. 5–7, 15, 21, 23, 25, 26, 38, 44, 47, 150).

[49] George Grätzer. *General Lattice Theory*. 2nd ed. Birkhäuser Verlag, 1998 (cit. on p. 18).

[50] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. “Just the Right Amount: Extracting Modules from Ontologies”. In: *Proceedings of the 16th International Conference on World Wide Web*. (Banff, Alberta, Canada). Ed. by Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy. ACM, May 2007, pp. 717–726 (cit. on p. 57).

[51] Jean-Luc Guigues and Vincent Duquenne. “Famille minimale d’implications informatives résultant d’un tableau de données binaires”. In: *Mathématiques et Sciences Humaines* 95 (1986), pp. 5–18 (cit. on pp. 6, 32).

[52] Volker Haarslev and Ralf Möller. “RACER System Description”. In: *Proceedings of the First International Joint Conference on Automated Reasoning*. (Siena, Italy). Ed. by Rajeev Goré, Alexander Leitsch, and Tobias Nipkow. Vol. 2083. Lecture Notes in Computer Science. Springer, June 2001, pp. 701–706 (cit. on p. 3).

[53] Mohamed Rouane Hacene, Marianne Huchard, Amedeo Napoli, and Petko Valtchev. "A Proposal for Combining Formal Concept Analysis and Description Logics for Mining Relational Data". In: *Proceedings of the 5th International Conference on Formal Concept Analysis*. (Clermont-Ferrand, France). Ed. by Sergei O. Kuznetsov and Stefan Schmidt. Vol. 4390. Lecture Notes in Computer Science. Springer, Feb. 2007, pp. 51–65 (cit. on p. 10).

[54] Mohamed Rouane Hacene, Marianne Huchard, Amedeo Napoli, and Petko Valtchev. "Relational concept analysis: mining concept lattices from multi-relational data". In: *Annals of Mathematics and Artificial Intelligence* 67.1 (2013), pp. 81–108 (cit. on p. 10).

[55] Petr Hájek. *The Metamathematics of Fuzzy Logic*. Kluwer, 1998 (cit. on p. 11).

[56] Frank van Harmelen, Vladimir Lifschitz, and Bruce Porter, eds. *Handbook of Knowledge Representation*. Elsevier Science, 2008 (cit. on p. 1).

[57] Hartmut von Hentig. *Magier oder Magister? Über die Einheit der Wissenschaft im Verständigungsprozeß*. Klett, 1972 (cit. on p. 5).

[58] Jochen Hipp, Ulrich Güntzer, and Gholamreza Nakhaeizadeh. "Algorithms for Association Rule Mining – A General Survey and Comparison". In: *SIGKDD Explorations* 2.2 (2000), pp. 1–58 (cit. on p. 101).

[59] Pascal Hitzler, Markus Krötzsch, and Sebastian Rudolph. *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC, 2009 (cit. on p. 83).

[60] Bernhard Hollunder, Werner Nutt, and Manfred Schmidt-Schauß. "Subsumption Algorithms for Concept Description Languages". In: *Proceedings of the 9th European Conference on Artificial Intelligence*. (Stockholm, Sweden). Ed. by João P. Martins and Michael Reinfrank. Lecture Notes in Computer Science. Springer, Aug. 1990, pp. 348–353 (cit. on p. 3).

[61] Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. "From $\mathcal{SHIQ}$ and RDF to OWL: The Making of a Web Ontology Language". In: *Journal of Web Semantics* 1.1 (2003) (cit. on p. 3).

[62] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. "Practical Reasoning for Very Expressive Description Logics". In: *Logic Journal of the IGPL* 8.3 (2000), pp. 239–263 (cit. on p. 3).

[63] Roni Khardon. "Translating between Horn Representations and their Characteristic Models". In: *Journal of Artificial Intelligence Research* 3 (1995), pp. 349–372 (cit. on p. 32).

[64] Boris Konev, Carsten Lutz, and Frank Wolter. "Exact Learning of TBoxes in $\mathcal{EL}$ and DL-Lite". In: *Informal Proceedings of the 26th International Workshop on Description Logics*. (Ulm, Germany). Ed. by Thomas Eiter, Birte Glimm, Yevgeny Kazakov, and Markus Krötzsch. Vol. 1014. CEUR Workshop Proceedings. CEUR-WS.org, July 2013, pp. 341–352 (cit. on p. 11).

[65] Sergei O. Kuznetsov. “Machine Learning and Formal Concept Analysis”. In: *Proceedings of the Second International Conference on Formal Concept Analysis*. (Sydney, Australia). Ed. by Peter W. Eklund. Vol. 2961. Lecture Notes in Computer Science. Springer, Feb. 2004, pp. 287–312 (cit. on p. 5).

[66] Sergei O. Kuznetsov. “On the Intractability of Computing the Duquenne-Guigues Base”. In: *Journal of Universal Computer Science* 10.8 (2004), pp. 927–933 (cit. on p. 35).

[67] Thomas Lukasiewicz and Umberto Straccia. “Managing Uncertainty and Vagueness in Description Logics for the Semantic Web”. In: *Journal of Web Semantics* 6.4 (2008), pp. 291–308 (cit. on p. 11).

[68] Michael Luxenburger. “Implications partielles dans un contexte”. In: *Mathématiques, Informatique et Sciences Humaines* 29.113 (1991), pp. 35–55 (cit. on pp. 10, 84, 92).

[69] Michael Luxenburger. “Implikationen, Abhängigkeiten und Galois-Abbildungen”. PhD thesis. TH Darmstadt, 1993 (cit. on pp. 10, 84, 92, 189).

[70] David Maier. *The Theory of Relational Databases*. Computer Science Press, 1983 (cit. on pp. 6, 31, 32).

[71] Marvin Minsky. *A Framework for Representing Knowledge*. Tech. rep. Memo 306. MIT-AI Laboratory, June 1974 (cit. on p. 3).

[72] Bernhard Nebel. “Computational Complexity of Terminological Reasoning in BACK”. In: *Artificial Intelligence* 34.3 (1988), pp. 371–383 (cit. on p. 3).

[73] Bernhard Nebel. “Terminological Cycles: Semantics and Computational Properties”. In: *Principles of Semantic Networks*. Morgan Kaufmann, 1991, pp. 331–362 (cit. on p. 58).

[74] Rafael Peñaloza Nyssen. “Axiom Pinpointing in Description Logics and Beyond”. PhD thesis. Technische Universität Dresden, 2009 (cit. on p. 57).

[75] Sergei A. Obiedkov and Vincent Duquenne. “Attribute-incremental construction of the canonical implication basis”. In: *Annals of Mathematics and Artificial Intelligence* 49.1-4 (2007), pp. 77–99 (cit. on pp. 7, 43).

[76] Silke Pollandt. “Datenanalyse mit Fuzzy–Begriffen”. In: *Begriffliche Wissensverarbeitung: Methoden und Anwendungen*. Ed. by Gerd Stumme and Rudolf Wille. Springer, 2000 (cit. on p. 11).

[77] Susanne Prediger. “Terminologische Merkmalslogik in der Formalen Begriffsanalyse”. In: *Begriffliche Wissensverarbeitung: Methoden und Anwendungen*. Ed. by Gerd Stumme and Rudolf Wille. Springer, 2000, pp. 99–124 (cit. on pp. 5, 10, 71).

[78] Susanne Prediger and Gerd Stumme. "Theory-driven Logical Scaling: Conceptual Information Systems meet Description Logics". In: *Proceedings of the 6th International Workshop on Knowledge Representation meets Databases*. (Linköping, Sweden). Ed. by Enrico Franconi and Michael Kifer. Vol. 21. CEUR Workshop Proceedings. CEUR-WS.org, July 1999, pp. 46–49 (cit. on p. 10).

[79] Susanne Prediger and Rudolf Wille. "The Lattice of Concept Graphs of a Relationally Scaled Context". In: *Conceptual Structures: Standards and Practices; Proceedings of the 7th International Conference on Conceptual Structures*. (Blacksburg, Virginia, USA). Ed. by William M. Tepfenhart and Walling R. Cyre. Vol. 1640. Lecture Notes in Computer Science. Springer, July 1999, pp. 401–414 (cit. on p. 9).

[80] Alan L. Rector, William Anthony Nowlan, and Galen Consortium. "The GALEN project". In: *Computer Methods and Programs in Biomedicine* 45.1-2 (1994), pp. 75–78 (cit. on p. 3).

[81] Sebastian Rudolph. "Exploring Relational Structures Via $\mathcal{FLE}$". In: *Conceptual Structures at Work: Proceedings of the 12th International Conference on Conceptual Structures*. (Huntsville, Alabama, USA). Ed. by Karl Erich Wolff, Heather D. Pfeiffer, and Harry S. Delugach. Vol. 3127. Lecture Notes in Computer Science. Springer, July 2004, pp. 196–212 (cit. on p. 9).

[82] Sebastian Rudolph. "Relational exploration: combining description logics and formal concept analysis for knowledge specification". PhD thesis. Technische Universität Dresden, 2006 (cit. on pp. 5, 8, 9).

[83] Sebastian Rudolph. "Some Notes on Pseudo-closed Sets". In: *Proceedings of the 5th International Conference on Formal Concept Analysis*. (Clermont-Ferrand, France). Ed. by Sergei O. Kuznetsov and Stefan Schmidt. Vol. 4390. Lecture Notes in Computer Science. Springer, Feb. 2007, pp. 151–165 (cit. on p. 43).

[84] Uwe Ryssel, Felix Distel, and Daniel Borchmann. "Fast algorithms for implication bases and attribute exploration using proper premises". In: *Annals of Mathematics and Artificial Intelligence* Special Issue 65 (2013), pp. 1–29 (cit. on p. 7).

[85] Klaus Schild. "A Correspondence Theory for Terminological Logics: Preliminary Report". In: *Proceedings of the 12th International Joint Conference on Artificial Intelligence*. (Sydney, Australia). Ed. by John Mylopoulos and Raymond Reiter. Morgan Kaufmann, Aug. 1991, pp. 466–471 (cit. on pp. 3, 52).

[86] Manfred Schmidt-Schauß. "Subsumption in KL-ONE is Undecidable". In: *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning*. (Toronto, Canada). Ed. by Ronald J. Brachman, Hector J. Levesque, and Raymond Reiter. Morgan Kaufmann, May 1989, pp. 421–431 (cit. on p. 3).

[87] Manfred Schmidt-Schauß and Gert Smolka. "Attributive Concept Descriptions with Complements". In: *Artificial Intelligence* 48.1 (1991), pp. 1–26 (cit. on pp. 3, 9).

[88] Barış Sertkaya. "Formal Concept Analysis Methods for Description Logics". PhD thesis. Technische Universität Dresden, 2007 (cit. on p. 9).

[89] Barış Sertkaya. "OntoComP: A Protégé Plugin for Completing OWL Ontologies". In: *The Semantic Web: Research and Applications; Proceedings of the 6th European Semantic Web Conference*. (Heraklion, Crete, Greece). Ed. by Lora Aroyo et al. Vol. 5554. Lecture Notes in Computer Science. Springer, May 2009, pp. 898–902 (cit. on p. 9).

[90] Evren Sirin, Bijan Parsia, Bernardo Grau, Aditya Kalyanpur, and Yarden Katz. "Pellet: A practical OWL-DL reasoner". In: *Journal of Web Semantics* 5.2 (2007), pp. 51–53 (cit. on p. 3).

[91] John Florian Sowa. *Principles of Semantic Networks*. Morgan Kaufmann, 1991 (cit. on p. 3).

[92] Gerd Stumme. "Attribute Exploration with Background Implications and Exceptions". In: *Data Analysis and Information Systems*. Ed. by Hans-Hermann Bock and Wolfgang Polasek. Studies in Classification, Data Analysis, and Knowledge Organization. Heidelberg: Springer, 1996, pp. 457–469 (cit. on pp. 8, 32, 47).

[93] Gerd Stumme. "Concept Exploration - A Tool for Creating and Exploring Conceptual Hierarchies". In: *Conceptual Structures: Fulfilling Peirce's Dream; Proceedings of the Fifth International Conference on Conceptual Structures*. (Seattle, Washington, USA). Ed. by Dickson Lukose, Harry S. Delugach, Mary Keeler, Leroy Searle, and John F. Sowa. Vol. 1257. Lecture Notes in Computer Science. Springer, Aug. 1997, pp. 318–331 (cit. on p. 8).

[94] Gerd Stumme. "Distributive Concept Exploration - A Knowledge Acquisition Tool in Formal Concept Analysis". In: *Proceedings of the 22nd Annual German Conference on Artificial Intelligence*. (Bremen, Germany). Ed. by Otthein Herzog and Andreas Günter. Vol. 1504. Lecture Notes in Computer Science. Springer, Sept. 1998, pp. 117–128 (cit. on p. 9).

[95] Gerd Stumme. "The Concept Classification of a Terminology Extended by Conjunction and Disjunction". In: *Proceedings of the 4th Pacific Rim International Conference on Artificial Intelligence*. (Cairns, Australia). Ed. by Norman Y. Foo and Randy Goebel. Vol. 1114. Lecture Notes in Computer Science. Springer, Aug. 1996, pp. 121–131 (cit. on p. 9).

[96] Gerd Stumme, Rafik Taouil, Yves Bastide, Nicolas Pasquier, and Lotfi Lakhal. "Computing iceberg concept lattices with TITANIC". In: *Data & Knowledge Engineering* 42.2 (2002), pp. 189–222 (cit. on p. 191).

[97] Gerd Stumme, Rafik Taouil, Yves Bastide, Nicolas Pasquier, and Lotfi Lakhal. "Intelligent Structuring and Reducing of Association Rules with Formal Concept Analysis". In: *KI 2001: Advances in Artificial Intelligence, Proceedings of the Joint German/Austrian Conference on AI*. (Vienna, Austria). Ed. by Franz Baader, Gerhard Brewka, and Thomas Eiter. Vol. 2174. Lecture Notes in Computer Science. Springer, Sept. 2001, pp. 335–350 (cit. on pp. 10, 84, 92, 95, 191).

[98] Alfred Tarski. "A Lattice-Theoretical Fixpoint Theorem and Its Applications". In: *Pacific Journal of Mathematics* (1955) (cit. on p. 59).

[99] Dmitry Tsarkov and Ian Horrocks. "FaCT++ Description Logic Reasoner: System Description". In: *Proceedings of the Third International Joint Conference on Automated Reasoning*. (Seattle, Washington, USA). Ed. by Ulrich Furbach and Natarajan Shankar. Vol. 4130. Lecture Notes in Computer Science. Springer, Aug. 2006, pp. 292–297 (cit. on p. 3).

[100] Anni-Yasmin Turhan. "On the Computation of Common Subsumers in Description Logics". PhD thesis. Technische Universität Dresden, 2007 (cit. on p. 57).

[101] Marcel Wild. "A Theory of Finite Closure Spaces Based on Implications". In: *Advances in Mathematics* 108.1 (1994), pp. 118–139 (cit. on p. 6).

[102] Rudolf Wille. *Begriffsdenken: Von der griechischen Philosophie bis zur Künstlichen Intelligenz*. FB4-Preprint Nr. 1724, TH Darmstadt. 1995 (cit. on p. 5).

[103] Rudolf Wille. "Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts". In: *Ordered sets* (1982). Ed. by Ivan Rival, pp. 445–470 (cit. on pp. 5, 6).

[104] Lotfi A. Zadeh. "Fuzzy Sets". In: *Information and Control* 8.3 (1965), pp. 338–353 (cit. on p. 11).

[105] Mohammed Javeed Zaki and Mitsunori Ogihara. "Theoretical Foundation of Association Rules". In: *Proceedings of the SIGMOD'98 Workshop on Research Issues in Data Mining and Knowledge Discovery*. 1998, pp. 1–8 (cit. on pp. 5, 101, 192).

[106] Monika Zickwolff. "Rule Exploration: First Order Logic in Formal Concept Analysis". PhD thesis. TH Darmstadt, 1991 (cit. on p. 8).

Affirmation

i. Hereby I affirm that I wrote the present thesis without any inadmissible help by a third party and without using any other means than indicated. Thoughts that were taken directly or indirectly from other sources are indicated as such. This thesis has not been presented to any other examination board in this or a similar form, neither in this nor in any other country.

ii. The present thesis has been written since September 2009 at the Institute of Algebra, Department of Mathematics, Faculty of Science, TU Dresden under the supervision of Prof. Bernhard Ganter.

iii. There have been no prior attempts to obtain a PhD at any university.

iv. I accept the requirements for obtaining a PhD (Promotionsordnung) of the Faculty of Science of the TU Dresden, issued February 23, 2011 with the changes in effect since June 15, 2011.

Versicherung

i. Hiermit versichere ich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

ii. Die vorliegende Dissertation wurde seit September 2009 am Institut für Algebra, Fachrichtung Mathematik, Fakultät Mathematik und Naturwissenschaften, Technische Universität Dresden unter der Betreuung von Prof. Bernhard Ganter angefertigt.

iii. Es wurden zuvor keine Promotionsvorhaben unternommen.

iv. Ich erkenne die Promotionsordnung der Fakultät Mathematik und Naturwissenschaften der TU Dresden vom 23. Februar 2011, in der geänderten Fassung mit Gültigkeit vom 15. Juni 2011 an.

Dresden, September 15, 2014