PRACTICAL REASONING FOR DEFEASIBLE DESCRIPTION LOGICS

by

Kody Moodley

Submitted in fulfilment of the academic requirements for the degree of

Doctor of Philosophy

in the

School of Mathematics, Statistics and Computer Science

University of KwaZulu-Natal,
Durban,
South Africa

Supervisor: **Prof. Thomas Meyer**
Co-Supervisors: **Prof. Uli Sattler, Dr. Deshen Moodley**

November 2015

Preface

The work reported on in this thesis was carried out at the Centre for Artificial Intelligence Research and the CSIR Meraka Institute in Pretoria, South Africa, from August 2011 to December 2015, under the supervision of Professor Thomas Meyer and Dr Deshen Moodley.

During the period from September 2013 to September 2014, the work was conducted at the Information Management Group of the department of Computer Science at the University of Manchester, United Kingdom, under the supervision of Professor Uli Sattler.

These studies represent original work by the author and have not otherwise been submitted in any form for any degree or diploma to any tertiary institution. Whenever other researchers work is used, they are duly acknowledged in the text.

Declaration 1 - Plagiarism

I, _____, declare that:

1. The research reported in this dissertation, except where otherwise indicated, is my original research.

2. This dissertation has not been submitted for any degree or examination at any other university.

3. This dissertation does not contain other person's data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.

4. This dissertation does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:

   (a) Their words have been re-written but the general information attributed to them has been referenced

   (b) Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.

5. This dissertation does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation and in the References section.

Signed: _____

Declaration 2 - Publications

Below is a list of publications contributing towards the content of this thesis.

1. Meyer, T., Moodley, K. and Varzinczak, I. (2012) A Protégé Plug-in for Defeasible Reasoning. In Proceedings of the Twenty Fifth International Workshop on Description Logics (DL). CEUR Workshop Proceedings, Volume 846, ISSN: 1613-0073.

2. Meyer, T., Moodley, K. and Varzinczak, I. (2012) A Defeasible Reasoning Approach for Description Logic Ontologies. In Proceedings of the Annual Research Conference of the South African Institute for Computer Scientists and Information Technologists (SAICSIT). p69-78, ISBN: 978-1-4503-1308-7, ACM New York.

3. Casini, G., Meyer, T., Moodley, K. and Varzinczak, I. (2013). Towards Practical Defeasible Reasoning for Description Logics. In Proceedings of the Twenty Sixth International Workshop on Description Logics (DL). CEUR Workshop Proceedings, Volume 1014, p587-599, ISSN: 1613-0073.

4. Casini, G., Meyer, T., Moodley, K. and Varzinczak, I. (2013). Nonmonotonic Reasoning in Description Logics: Rational Closure for the ABox. In Proceedings of the Twenty Sixth International Workshop on Description Logics (DL). CEUR Workshop Proceedings, Volume 1014, p600-615, ISSN: 1613-0073.

5. Casini, G., Meyer, T., Moodley, K. and Nortje, R. (2014). Relevant Closure: A New Form of Defeasible Reasoning for Description Logics. In Proceedings of the Fourteenth European Conference on Logics in Artificial Intelligence (JELIA). Lecture Notes in Computer Science

(LNCS), Volume 8761, p92-106, ISBN: 978-3-319-11557-3, Springer International Publishing.

6. Meyer, T., Moodley, K. and Sattler, U. (2014). Practical Defeasible Reasoning for Description Logics. In Proceedings of The Seventh European Starting AI Researcher Symposium (STAIRS). Volume 264, p191-200, ISBN: 978-1-61499-420-6, IOS Press.

7. Meyer, T., Moodley, K. and Sattler, U. (2014). DIP: A Defeasible-Inference Platform for OWL Ontologies. In Proceedings of the Twenty Seventh International Workshop on Description Logics (DL). CEUR Workshop Proceedings, Volume 1193, p671-683, ISSN: 1613-0073.

8. Casini, G., Meyer, T., Moodley, K., Sattler, U. and Varzinczak, I. (2015). Introducing Defeasibility into OWL Ontologies. In Proceedings of the International Semantic Web Conference (ISWC). Lecture Notes in Computer Science (LNCS), Volume 9367, p409-426, ISBN: 978-3-319-25009-0, Springer International Publishing.

I was the primary author for Publications 1 to 3 and 6 to 8 in which I contributed somewhat to the theoretical aspects but more heavily in practical implementation and evaluation of tools and algorithms. For publications 4 and 5, I was responsible for the implementation, experimental evaluation (and write-up thereof) of several algorithms central to the work.

Signed: _____

Acknowledgements

Firstly, I would like to thank my supervisors Tommie Meyer, Uli Sattler and Deshen Moodley. I really could not have asked for better supervisors and they were each integral to completing this PhD. Thank you also to Giovanni Casini and Ivan Varzinczak for helpful discussions on my work.

I am very grateful that, during my tenure at the University of Manchester, Uli and the students of the Information Management Group really went out of their way to make me feel welcome. In particular, I would like to thank Slava Sazonau, Liang Chang, Nico Matentzoglu and Jared Leo for keeping me sane and entertained while I conducted my research. Thank you to Nico for helping me with sourcing data for the empirical component of my work, and thanks also to Bijan Parsia for his useful comments on my experiments. Matthew Horridge was an inspiration to me on how to conduct practical research in my area and I am grateful to him for his example in this regard.

Back in Pretoria, I would like to thank Marlene Jivan for all her help with handling various administration tasks relevant to this PhD, and Nishal Morar for keeping me company at the office while he did his Masters degree. Of course, this PhD could not have been completed without funding and I have been very fortunate to receive it in abundance from the National Research Foundation, the Commonwealth Scholarship Commission, the CSIR Meraka Institute and the University of KwaZulu-Natal. I would particularly like to thank Quentin Williams from the CSIR Meraka Institute for his help with arranging funds for me.

Finally, I would like to thank my family for their encouragement during my studies, and a very special appreciation goes to my wife Nicole for her unconditional love, understanding and support throughout this PhD - thanks for putting up with me.

Contents

List of Figures

List of Algorithms

Abstract

Description Logics (DLs) are a family of logic-based languages for formalising ontologies. They have useful computational properties allowing the development of automated reasoning engines to infer implicit knowledge from ontologies. However, classical DLs do not tolerate exceptions to specified knowledge. This led to the prominent research area of *nonmonotonic* or *defeasible* reasoning for DLs, where most techniques were adapted from seminal works for propositional and first-order logic.

Despite the topic's attention in the literature, there remains no consensus on what "sensible" defeasible reasoning means for DLs. Furthermore, there are solid foundations for several approaches and yet no serious implementations and practical tools. In this thesis we address the aforementioned issues in a broad sense. We identify the *preferential* approach, by Kraus, Lehmann and Magidor (KLM) in propositional logic, as a suitable abstract framework for defining and studying the precepts of sensible defeasible reasoning.

We give a generalisation of KLM's precepts, and their arguments motivating them, to the DL case. We also provide several preferential algorithms for defeasible entailment in DLs; evaluate these algorithms, and the main alternatives in the literature, against the agreed upon precepts; extensively test the performance of these algorithms; and ultimately consolidate our implementation in a software tool called *Defeasible-Inference Platform* (DIP).

We found some useful entailment regimes within the preferential context that satisfy *all* the KLM properties, and some that have scalable performance in real world ontologies even without extensive optimisation.

Chapter 1

Introduction

An ontology is a knowledge representation artefact which, either formally or informally, specifies a conceptual understanding of a domain of interest.

The main conceptual building blocks of ontologies are *individuals* (objects in the domain), *concepts* (classes of objects in the domain) and *roles* (relationships between objects, or classes of objects, in the domain).

Description Logics (DLs) [10] are a family of logic-based languages (fragments of First-order logic) traditionally used for representing ontologies. With DLs one can formalise the aforementioned components of ontologies, as well as specify logical relationships between them, using logical sentences called DL *axioms*. DL ontologies can thus be thought of, in a sense, as sets of DL axioms describing a domain of interest. DL ontologies are the types we are exclusively interested in for the purposes of this thesis, and therefore, all further references to the term ontology in this thesis allude to these.

Key features of DLs themselves which make them interesting from a practical perspective are (1) they have a precise, logic-based syntax and model-theoretic semantics which eliminates ambiguity in represented knowledge and (2) they are decidable, meaning that one can devise procedures for automated inference from DL-specified knowledge, that are guaranteed to terminate.

The decidability of DLs eventually led to the development of highly-optimised practical procedures for performing inference on-demand, often in

a matter of micro seconds, for real-world application ontologies. Their success as formalisms for representing application ontologies is attested by their service as the logical underpinning of the Web Ontology Language (OWL) [101]. OWL is a set of languages for formalising ontologies on the Web towards the realisation of Tim Berners-Lee's vision of the *Semantic Web* [25].

OWL became a W3C recommendation for representing ontologies on the Web in 2004, and is currently in its second major version - OWL 2 [84].

1.1 Motivation

A limitation of classical DLs is their inability to cope with *exceptions*. That is, their inability to represent and reason with knowledge that is generally sound but permits exceptions. For example, while the proposition that students *usually* do not pay taxes appears to be a sound one, it is quite possible that there are exceptional types of student who *are* required to pay taxes - for example *employed* students.

Classical DLs will not allow this common sense reasoning because all stated facts are considered "universally true" or infallible. Therefore, we are only allowed to state rigid statements like "students do not pay taxes", and such information has to be logically reconciled with any additional information we discover. This behaviour is certainly useful for many applications.

For example, adding to our knowledge that employed students are *types* of student, classical DLs have to infer that employed students inherit the property of general students, and thus do not pay taxes. Again, this is very sensible reasoning provided we have no other information.

However, if we later discover that employed students are required to pay taxes, while intuition might tell us that employed students are *exceptional* students, classical DLs will have to reconcile this new information with what we knew previously - that employed students are exempt from taxes. The only way to logically reconcile these incompatible situations in classical DLs is to infer that employed students do not exist. Clearly, this is not suitable

reasoning behaviour in contexts where we would like to express generalised (defeasible) statements and permit exceptions to these statements without leading to logical incoherences and other incompatibilities.

Within the area of reasoning with exceptions in DLs, we have found that there is a general scarcity of practical implementations and tools. In addition, while it is clear that not all theoretical issues have been resolved in the majority of proposals, we believe that in many cases the foundation is solid enough to begin practical investigations. Such practical investigations are not meant purely for evaluating reasoning performance to complement existing complexity results. Rather, the impetus we are advocating to address practical issues in defeasible reasoning, is meant to lead to concrete implementations and tools. These tools, in turn, are meant for studying the behaviour of reasoning proposals in practical settings to gain deeper insight into their inferential merit. User feedback from such tools would then drive the refinement of the theoretical notions in defeasible reasoning.

In other words we highlight that, unlike in many classical logics, defeasible reasoning has no universally agreed upon definition for entailment. We therefore believe that, while there are guidelines on what constitutes sensible defeasible reasoning, there is not necessarily a *unique* notion of "sensible" for a particular formalism. Therefore, it is ultimately up to the user to decide what inferential behaviour suits them best in different applications.

Existing approaches to addressing exceptions in DLs have mainly descended from five methodologies: Circumscription [136], Default Logic [161], Preferential Logic [117], Autoepistemic Logic [141] and Defeasible Logic [149]. We give more detailed presentations of applicable descendents of these formalisms in Chapter 2. However, since Autoepistemic Logic does not address exceptions in a direct way, and since the rule-based approach of Defeasible Logic is quite orthogonal to the aforementioned formalisms, the descendents of these two approaches are not given as much attention in this thesis.

1.2 Uncertainty vs. Exceptions in Knowledge Representation

It must be mentioned that there are a variety of "modes" of defeasibility that can be introduced into KR formalisms. In Section 2.3 we argue that the root cause of the need for defeasible representation is the lack of access to complete knowledge. This incompleteness of knowledge leads to multifarious modes of defeasible representation. A popular term often used to collectively refer to these modes is "uncertainty". However, we believe that the term "uncertainty" could be misleading if we use it to describe the notion of defeasibility that we are interested in for this thesis.

The statements "john is either a quaker or a pacifist", "there is a 25% chance of snow tomorrow" and "it is possible that susan went home" all have symptoms of uncertainty in their representations. Therefore, the term uncertainty is quite appropriate to describe these representations.

However, for this thesis, we are interested in representing *generalisations* that hold in typical, usual, normal or default circumstances, but that permit the occurrence of exceptions. For example, we would like to be able to state that students *usually* do not pay taxes or *typical* sushi dishes contain fish.

Upon encountering exceptions, such as employed students or vegetarian varieties of sushi, we would still like our reasoning to be able to conclude meaningful knowledge about sushi and students. Therefore, we do not use the term uncertainty to describe the issue we are dealing with. Rather, we use the term *exceptions* which more accurately describes our problem of focus.

1.3 Goals

The broad goal of this thesis is to select an appropriate formalism from the literature and take the first steps towards making it practical for DLs. We choose to extend the *preferential* approach [111, 117, 116], by Kraus, Lehmann and Magidor (KLM), towards this aim.

The main reason for choosing the preferential approach is that it has a much broader and abstract perspective on defeasible reasoning. In particular, it pioneered the study of nonmonotonic inference from the perspective of the consequence relations that it could induce. This led to the formulation of a series of inference rules (called the KLM postulates) that KLM argue any "rational" defeasible reasoning mechanism should satisfy.

We believe that such an abstract view is very helpful in developing "sound" and "sensible" defeasible reasoning behaviour (more details are presented in Chapter 2). In fact, one of the goals of this thesis is to motivate the logical merit of the KLM postulates within the setting of DLs.

That is to say, we address the question: since defeasible reasoning allows one to draw "plausible" inferences from "generally" sound knowledge, and there are no universally agreed upon definitions for "plausible" and "general", what common rules of inference should *any* reasonable definition satisfy?

In the same thread we plan to *evaluate* the main existing approaches to defeasible reasoning, as well as novel preferential algorithms introduced by us, against these rules of inference. This would give us a clearer picture as to the inferential character of each proposal.

Towards practically implementing preferential reasoning for DLs, we are going to address the following general questions:

(1) Is there actually a demonstrable need for defeasible reasoning in DL-based ontologies (in real-world application settings)? We provide quantitative evidence for this because there are only anecdotal and subjective arguments for this in the literature.

(2) What are some useful defeasible entailment regimes within the preferential framework? (3) Which of our preferential algorithms (and the alternative formalisms in the literature) satisfy the KLM postulates? (4) If some of the regimes or formalisms do not satisfy all the postulates, which ones do they not satisfy? (5) How does the practical performance of the preferential algorithms look? (6) Is it possible to integrate preferential reasoning algorithms into existing tools for ontology development? (7) If so, how can this

be done in a fairly unobtrusive and seamless manner? Of course, there are many interesting sub-questions to each of the aforementioned questions, and we explore some of these in the rest of this thesis.

1.4 Organisation of Thesis

The thesis is structured as follows: Chapter 2 gives some background, logical preliminaries and related work on defeasible reasoning for DLs.

Chapter 2, in giving a flavour for the current defeasible reasoning proposals in DLs, constitutes solid ground from which to compare the representational aspects of each formalism, as well as their entailment mechanisms.

In Section 2.8 we introduce the preferential reasoning approach for DLs which constitutes the theoretical foundation of this thesis. This is because we actually select this approach as our formalism of choice to extend towards the pragmatic goals of this thesis. We motivate why we choose this formalism over the others in Section 2.10.

In Chapter 3 we again take a more general perspective than just preferential reasoning. There we perform an experimental evaluation to give quantitative evidence suggesting that defeasible reasoning is indeed required in real world application ontologies. This strengthens the anecdotal arguments in the literature from the point-of-view of biology and biomedicine.

We also extend KLM's motivational arguments for their rationality postulates by giving a generalisation of these postulates to the DL case, and arguing for their logical merit in this setting. We show, by means of examples, that these postulates capture intuitive rules of inference that any defeasible reasoning mechanism for DLs should satisfy.

Lehmann also argues the case for the rationality of some *informal* properties of inference in the propositional setting. We generalise his arguments to the DL case also in Chapter 3.

Chapter 4 gives a detailed description of five algorithmic constructions for preferential reasoning in DLs. The chapter also makes explicit many elemen-

tary but interesting theoretical results about these constructions. We also give a detailed explanation as to how preferential reasoning can be reduced to classical reasoning for DLs.

Chapter 5 evaluates our preferential reasoning algorithms, and the main alternative formalisms of Chapter 2, against the *formal* KLM postulates. This gives us a general picture of the inferential characters of the main existing defeasible reasoning approaches for DLs.

In Chapter 6 we perform a thorough performance evaluation of our preferential reasoning algorithms. Since naturally occurring data is not available, we take two approaches to sourcing data for the evaluation: a purely synthetic approach, as well as a way to introduce defeasible subsumption into existing classical ontologies.

Chapter 7 presents the implementation aspects of a tool we have developed called Defeasible-Inference Platform (DIP). The tool is integrated into the well-known ontology editor Protégé and is able to execute all the algorithms presented in Chapter 4. Finally, we give our conclusions, summaries and directions for future work in Chapter 8.

Chapter 2

Background and Related Work

In order to place past work in the area into context, it seems prudent to refer back to the motivating problem(s) which ushered investigation into so-called *nonmonotonic* or *defeasible* logics.

One of the "holy grails" of Artificial Intelligence (AI) in general is to be able to simulate or automate the reasoning methodology of a rational human being. Indeed, the sub-area of AI called Knowledge Representation and Reasoning (KRR) is directly concerned with the development of appropriate languages (often rooted in formal logic and given a formal semantics) to represent knowledge.

The knowledge itself usually takes its form as a set of logical sentences describing the chosen topic of interest called a *formal ontology* or *knowledge base* (KB for short), along with an entailment relation and accompanying procedures for deriving inferences from this knowledge.

One of the defining characteristics of classical logics for ontology specification is that they comply with the property of *monotonicity*. A monotonic formalism enforces that *addition* of knowledge to the KB can only result in *additional* inferences. In other words, addition of new information that conflicts with the old information cannot force "overriding" or ignorance of the old information. Essentially, this property models the implicit assumption that all facts in the KB are universally true or infallible. One can immediately

see that this property is unsuitable for modelling knowledge which is subject to *exceptions*. Human beings are notably adept at performing rational inference in the presence of exceptions.

Students are exempt from paying tax, unless they are not when they are employed [72]. Sushi contains fish, unless it does not in the case of vegetarian varieties. Birds fly, unless they do not in the case of ostriches and penguins. From a theoretical perspective, this is the core problem that the related approaches listed in this chapter address, including the approach that we base our own investigation on. That is, how to enable computer systems to perform similarly well in deriving rational inferences from knowledge, in the presence of exceptions.

It must be noted that the list of approaches to reasoning with exceptions given in this chapter is not an exhaustive one. Rather, these are the main (or more well-known) approaches to this problem in the literature. For various reasons mentioned in Chapter 1, we have chosen DLs as our formalisms to focus on. Therefore, in this chapter, we survey only those related approaches that address DLs.

In Section 2.1, we present the syntax and semantics of classical DLs. In Sections 2.4 to 2.9 we give a concise account of the most popular approaches addressing exceptions specifically within the context of DLs. We very briefly compare these approaches in Section 2.10 (although we have a more detailed comparison of their inferences in Chapter 5). Finally, in Section 2.11, we very briefly mention some relevant notation, acronyms and terminology that we adopt throughout this thesis.

2.1 Description Logics (DLs)

Description Logics [10] are a family of fragments of first-order logic that have good computational properties (They are all decidable). DLs are used for representing declarative knowledge about a chosen domain of interest, while enabling automated reasoning over such knowledge to reveal implicit

facts about the domain. DLs also form the logical underpinning of the Web Ontology Language (OWL) which is the W3C recommended formalism for representing ontologies on the Semantic Web [25].

Each DL offers a set of logical features which together determines the expressive power of that particular DL, that is, its scope for representing various types of knowledge. However, the more expressive the DL the more computationally intensive it is to perform automated inference with it. This trade-off between expressivity and complexity should therefore be considered when picking the appropriate DL for a particular application.

In this thesis, our main focus will be on the well-known DL $\mathcal{ALC}$ [175] (and more expressive relatives), which is generally considered to be of moderate to high expressivity.

2.1.1 Syntax and Semantics

In this section, we present the basic expressive DL $\mathcal{ALC}$. $\mathcal{ALC}$ forms the basis, in later chapters, for our investigation into DL-extensions that are able to handle exceptions.

Syntax of $\mathcal{ALC}$

We start with a basic vocabulary: a set of concept names N_c (also called atomic concepts), role names N_r (also called atomic roles) and individual names N_i. N_c additionally contains the special concepts $\top$ (The top concept) and $\bot$ (The bottom concept), which we define in the next section. The core features of $\mathcal{ALC}$, namely negation ($\neg$), conjunction ($\sqcap$), disjunction ($\sqcup$), existential role quantification ($\exists$) and universal role quantification ($\forall$), can be combined with the basic vocabulary to form complex concept expressions.

Given a concept name $A \in N_c$ and a role name $R \in N_r$, the permissible combinations to produce a complex concept C can be specified inductively by:

$$C ::= A \mid \top \mid \bot \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \exists R.C \mid \forall R.C$$

To give some examples: let Human, Male, Red, Green and Blue be concept names, and let hasChild be a role name. We can denote the set of females in our domain by the expression Human$\sqcap\neg$Male using negation and conjunction. To refer to the objects in our domain that are fathers who have at least one daughter we can use the expression Human $\sqcap$ Male $\sqcap \exists$hasChild.$\neg$Male, which makes use of negation, conjunction and an existential role quantifier.

In general, for concepts of the form $\exists$R.C, we refer to C as a *role filler* for R in $\exists$R.C. Those people who only have sons can be captured by Human $\sqcap$ $\forall$hasChild.Male, using conjunction and a universal role quantifier. Using disjunction, we can refer to all the objects in our domain that have a primary colour: Red $\sqcup$ Green $\sqcup$ Blue.

Of course, since our concept grammar is inductively defined, we can nest expressions as well. For example, grandparents who only have granddaughters can be captured by $\exists$hasChild.($\exists$hasChild.$\top$) $\sqcap$ $\forall$hasChild.($\forall$hasChild.$\neg$Male).

In addition to the capability of forming concept expressions, the thrust of DLs is to represent knowledge in the form of logical sentences (called *axioms*) describing relationships between concept expressions. The main type of axiom in DLs is called *subsumption* or *inclusion*. Subsumptions have the form $C \sqsubseteq D$ (intuitively meaning that C is a subtype of D) where C and D are DL concepts. In the case where we do not restrict C and D to be concept names, we refer to such subsumptions as *general concept inclusions (GCIs)*. The expression $C \equiv D$ (intuitively meaning that C is equivalent to D) is called an *equivalence* statement and it abbreviates $\{C \sqsubseteq D, D \sqsubseteq C\}$. Given concepts C and D, axioms of the form $C \parallel D$ are called *disjointness axioms* (intuitively capturing that C and D are disjoint or distinct).

Examples of concept inclusions include: Heart $\sqsubseteq$ AnatomicalOrgan, Student$\sqcup$Lecturer $\sqsubseteq \exists$hasAccess.Library, Dog $\sqsubseteq \forall$hasClaw.NonRetractableClaw. Respectively, these subsumptions capture that: a heart is a type of anatomical organ, if one is either a student or a lecturer (or both) then one has access to the university library and dogs only have non-retractable claws.

Some examples of equivalence axioms include: $\mathsf{Pizza} \equiv \exists \mathsf{hasPart.PizzaBase}$ and $\mathsf{Lecturer} \equiv \exists \mathsf{worksFor.University} \sqcap \exists \mathsf{teaches.UniversityCourse}$. The former captures that pizzas have pizza bases and the "other direction" is also true i.e., things which have pizza bases are pizzas. Similarly, the latter axiom specifies that lecturers work for some university and teach some university course. The other direction seems intuitive as well, that is, entities who work for a university and teach a course are lecturers. Equivalence statements that have a concept name on one side of the symbol $\equiv$, are sometimes called *definitions* because they accomplish exactly that - defining terms in the KB (such as $\mathsf{Lecturer}$ and Pizza in our examples).

Some examples of disjointness statements include $\mathsf{Dog} \parallel \mathsf{Cat}$ and $\mathsf{Male} \parallel \mathsf{Female}$ where, intuitively, these capture that dogs and cats (and resp. males and females) are disjoint or distinct entities (one cannot be both). As we shall discover later, disjointness and equivalence statements can be captured by using subsumption and are thus just syntactic sugar for allowing syntactic abbreviations for axioms.

Therefore, a finite set of GCIs constitutes the so-called *terminological* or *intensional* component of an $\mathcal{ALC}$ knowledge base (KB) - *TBox* for short.

One can also assign properties to individual objects (or pairs of objects) from the domain using so-called *assertional* or *extensional* statements. There are two basic types of assertion: *concept assertions* of the form $\mathsf{C}(\mathsf{a})$ (where C is a possibly complex $\mathcal{ALC}$ concept), and *role assertions* of the form $\mathsf{R}(\mathsf{a}, \mathsf{b})$. $\mathsf{C}(\mathsf{a})$ intuitively states that a is an instance of C and $\mathsf{R}(\mathsf{a}, \mathsf{b})$ intuitively means that a is related to b through the role R.

Some examples of ABox assertions include: $\mathsf{Student}(john)$ (intuitively, $john$ is a student) and $\mathsf{marriedTo}(john, mary)$ ($john$ is married to $mary$). A finite set of assertions about our domain is called an *ABox* which forms the extensional component of the KB. Thus an $\mathcal{ALC}$ KB is a tuple $(\mathcal{T}, \mathcal{A})$ where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox. Either the ABox or TBox (or both) may be empty. In the next section we give the precise logical meaning of concept expressions as well as TBox and ABox statements.

Semantics of $\mathcal{ALC}$

In keeping with Tarskian-style semantics, the "truth" of an axiom in $\mathcal{ALC}$ is determined by its satisfaction in a relevant *interpretation*. First-order interpretations are adopted for the semantics of DLs and hence $\mathcal{ALC}$. An $\mathcal{ALC}$ interpretation $\mathcal{I}$ is a structure $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ where $\Delta^{\mathcal{I}}$ is a non-empty set of objects denoting the domain of discourse and $\cdot^{\mathcal{I}}$ is a called an *interpretation function*. The function maps each individual name $a \in N_i$ to an element of the domain ($a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$), each concept name $A \in N_c$ to a subset of the domain ($A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$) and each role name $R \in N_r$ to a set of *ordered* pairs on the elements of the domain ($R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$).

For complex concepts, $\cdot^{\mathcal{I}}$ is defined inductively as follows: *Let $\mathcal{I}$ be an interpretation, R be a role name and C, D $\mathcal{ALC}$ concepts, then*:

$$
\begin{aligned}
&\top^{\mathcal{I}} = \Delta^{\mathcal{I}} \\
&\bot^{\mathcal{I}} = \emptyset \\
&(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}} \\
&(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
&(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
&(\exists R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \text{there is a } b \in C^{\mathcal{I}} \text{ s.t. } (a,b) \in R^{\mathcal{I}}\} \\
&(\forall R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \text{for all } b, (a,b) \in R^{\mathcal{I}} \implies b \in C^{\mathcal{I}}\}
\end{aligned}
$$

Similarly, for TBox and ABox statements, an interpretation $\mathcal{I}$ satisfies:

$$
\begin{array}{llll}
C \sqsubseteq D & \text{if} & C^{\mathcal{I}} \subseteq D^{\mathcal{I}} & (C \textit{ is subsumed by } D), \\
C \equiv D & \text{if} & C^{\mathcal{I}} = D^{\mathcal{I}} & (C \textit{ is equivalent to } D), \\
C \parallel D & \text{if} & C^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset & (C \textit{ is disjoint with } D), \\
C(a) & \text{if} & a^{\mathcal{I}} \in C^{\mathcal{I}} & (a \textit{ is an instance of } C), \\
R(a,b) & \text{if} & (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}} & (a \textit{ is related to } b \textit{ via } R)
\end{array}
$$

If an interpretation $\mathcal{I}$ satisfies an axiom α (written as $\mathcal{I} \Vdash \alpha$) then it is referred to as a *model* for that axiom. An interpretation $\mathcal{I}$ is a model for a TBox (resp. ABox) $\mathcal{T}$ (resp. $\mathcal{A}$) if it satisfies *all* the axioms in $\mathcal{T}$ (resp. $\mathcal{A}$).

These results can be written as $\mathcal{I} \Vdash \mathcal{T}$ (resp. $\mathcal{I} \Vdash \mathcal{A}$). Therefore, $\mathcal{I}$ will be a model for an $\mathcal{ALC}$ KB $(\mathcal{T}, \mathcal{A})$ if is a model for both $\mathcal{T}$ and $\mathcal{A}$.

2.1.2 Common Reasoning Problems

The precise syntax and semantics of DLs, together with their nice computational properties, make them amenable to automated inference. There are three basic types of reasoning tasks for DLs namely, *concept or role satisfiability*, *KB satisfiability* and the more general task of *entailment*.

Definition 1 (Concept and Role Satisfiability) *Let $\mathcal{K}$ be a DL KB and let C (resp. R) be a possibly complex concept (resp. role name). C (resp. R) is* satisfiable *w.r.t. $\mathcal{K}$ if there is a model $\mathcal{I}$ for $\mathcal{K}$ s.t. $C^{\mathcal{I}} \neq \emptyset$ (resp. $R^{\mathcal{I}} \neq \emptyset$).*

If a concept or role name is not satisfiable w.r.t. to a KB then we say it is *unsatisfiable* w.r.t. the KB.

Example 1 *Consider the following KB $\mathcal{K}$:*

$$
\begin{array}{lcl}
\mathsf{SweetTaste} & \sqsubseteq & \neg\mathsf{SavouryTaste}, \\
\mathsf{Fruit} & \sqsubseteq & \forall \mathsf{taste}.\mathsf{SweetTaste}, \\
\mathsf{Vegetable} & \sqsubseteq & \forall \mathsf{taste}.\mathsf{SavouryTaste}, \\
\mathsf{Tomato} & \sqsubseteq & \mathsf{Fruit} \sqcap \exists \mathsf{taste}.\mathsf{SavouryTaste}, \\
\mathsf{SweetPotato} & \sqsubseteq & \mathsf{Vegetable} \sqcap \exists \mathsf{taste}.\mathsf{SweetTaste}
\end{array}
$$

The concept names Tomato and SweetPotato, in Example 1 on Page 30, are unsatisfiable w.r.t. $\mathcal{K}$. That is, one cannot construct a model for $\mathcal{K}$ in which there are Tomato (or SweetPotato) objects. □

The task of checking satisfiability in DLs can be summed up as follows: given a KB $\mathcal{K}$ and a possibly complex concept C (or role name R), determine if C (or R) is satisfiable w.r.t. $\mathcal{K}$.
If there is at least one concept *name* or role name in a KB that is unsatisfiable (as is the case in $\mathcal{K}$) we say that the KB is *incoherent*. Incoherence

is generally considered to be an undesired or unintended situation in ontology engineering. In most cases, appropriate *debugging* techniques [174] are adopted to "repair" such consequences.

A more serious problem is if the KB is unsatisfiable (we also say that the KB is *inconsistent*), which leads us to define the circumstance under which a KB is satisfiable (or *consistent*).

Definition 2 (KB Satisfiability or Consistency) *A DL KB $\mathcal{K}$ is satisfiable (or* consistent*) if it has a model.*

Recall that Tomato is unsatisfiable w.r.t. $\mathcal{K}$ in Example 1 on Page 30. This enforced that in all models for $\mathcal{K}$ there could not be any instance of tomato. Suppose we now add the assertion $\mathsf{Tomato}(t1)$ to $\mathcal{K}$. This enforces that in *each* model for $\mathcal{K} \cup \{\mathsf{Tomato}(t1)\}$, there should be a tomato instance referred to as $t1$. This is a logical contradiction and means that there is no model for $\mathcal{K} \cup \{\mathsf{Tomato}(t1)\}$. This KB is therefore *inconsistent*. The DL consistency task is principally straightforward: determine if the given KB has a model.

Finally, the most general reasoning task of DLs is that of *entailment*. That is, determining if a particular ABox or TBox statement "logically follows" from a given KB.

Definition 3 (Entailment) *Given a DL KB $\mathcal{K}$ and a TBox (or ABox statement) α, α is* entailed by $\mathcal{K}$*, written as $\mathcal{K} \models \alpha$, if each model for $\mathcal{K}$ is also a model for α.*

An implicit semantic property of DLs is the *open-world assumption* (OWA), which is the assumption in DLs that information contained in our KB is incomplete. This is in contrast to the *closed-world assumption* (CWA) frequently used in databases [160]. Consider an $\mathcal{ALC}$ KB $(\mathcal{T}, \mathcal{A})$ where the TBox $\mathcal{T} = \emptyset$ and the ABox $\mathcal{A} =$
$\{\mathsf{hasIngredient}(dish1, carrot1)\}$. $\mathcal{A}$ mentions that there is a meal called $dish1$ which has an object called $carrot1$ as an ingredient. Using the CWA, one will derive that $carrot1$ is the *only* ingredient in the dish. In fact, if we additionally add to $\mathcal{A}$ the assertion $\mathsf{Vegetable}(carrot1)$, and to our TBox $\mathcal{T}$ the

axiom $\forall$hasIngredient.Vegetable $\sqsubseteq$ VegetarianDish, we can go so far as to obtain $(\mathcal{T}, \mathcal{A}) \models$ VegetarianDish($dish1$). On the other hand, if we use the OWA then we cannot infer that carrot is the only ingredient of $dish1$. All we know is that $dish1$ has carrot as one of its ingredients (it may have others that are not yet mentioned). The OWA allows for the possibility of more ingredients (not explicitly stated) and hence we cannot deduce VegetarianDish($dish1$). The OWA and CWA have their respective appropriate applications. DLs, as a direct result of the semantics, employ the OWA and this has proven to be appropriate for most DL applications. For more detailed information about these issues, the interested reader should consult the DL Handbook [10]. The tasks of determining concept or role satisfiability and KB consistency are very important in the ontology engineering process. Both tasks can be reduced to the task of entailment: a concept C is unsatisfiable w.r.t. a KB $\mathcal{K}$ if and only if $\mathcal{K} \models C \sqsubseteq \bot$ and $\mathcal{K}$ is inconsistent if and only if $\mathcal{K} \models \top \sqsubseteq \bot$. However, their most common use case (in their most literal sense) is to ensure that the ontology developed avoids becoming incoherent or inconsistent. I.e., these tasks are generally employed in an iterative manner on the KB, after some modifications are made to it. In contrast, the "bread and butter" of automated DL reasoning remains the derivation of implicit knowledge (called entailments) from KBs that are both *consistent* and *coherent*.

Example 2 *Consider the following KB $\mathcal{K}$ [8]:*

$$\begin{array}{lcl}
\exists \mathsf{systolic_pressure}.\mathsf{High_pressure} & \sqsubseteq & \exists \mathsf{finding}.\mathsf{Hypertension}, \\
\exists \mathsf{finding}.\mathsf{Hypertension} \sqcap \exists \mathsf{history}.\mathsf{Hypertension} & \sqsubseteq & \exists \mathsf{risk}.\mathsf{Myocardial_infarction}, \\
\mathsf{RiskyPatient} & \equiv & \exists \mathsf{risk}.\mathsf{Myocardial_infarction}, \\
\mathsf{systolic_pressure}(\mathsf{bob}, \mathsf{p1}), \ \mathsf{High_pressure}(\mathsf{p1}), & & \\
\mathsf{history}(\mathsf{bob}, \mathsf{h1}), \ \mathsf{Hypertension}(\mathsf{h1}) & &
\end{array}$$

In Example 2 on Page 32 we can deduce that Bob is at risk of getting a heart attack. I.e., $\mathcal{K} \models$ RiskyPatient(bob). In each model for $\mathcal{K}$ the object referred to as *bob* belongs to the extension RiskyPatient. □

Therefore, the problem of entailment is: given a KB $\mathcal{K}$ and an axiom α, determine if $\mathcal{K} \models \alpha$. The exhaustive or brute-force approach of constructing all possible models for $\mathcal{K}$, and examining the constraints that these models satisfy, is not viable because there are infinitely many of them.

Tableau-based algorithms avoid this naïve approach, evolving over the years with the help of numerous optimisations to the current maturity of having feasible performance for "on-demand" reasoning [98, 102, 103]. It can be shown that, for some DLs, the tasks of satisfiability, consistency and entailment can be reduced to each other.

For example, checking if $\mathcal{K} \models \mathsf{RiskyPatient}(\mathsf{bob})$ in Example 2 on Page 32 can be accomplished by adding $\neg\mathsf{RiskyPatient}(\mathsf{bob})$ to $\mathcal{K}$ and checking whether the resulting KB is consistent. If it is inconsistent then we can deduce that $\mathcal{K} \models \mathsf{RiskyPatient}(\mathsf{bob})$. Also, given a KB $\mathcal{K}$ and concepts C, D, $\mathcal{K} \models C \sqsubseteq D$ if and only if the concept $C \sqcap \neg D$ is unsatisfiable w.r.t. $\mathcal{K}$ [10].

In general, the preferred reduction in practice is reducing general entailment to the consistency problem. The task of consistency is concerned with determining if there is a model for a given KB. Therefore, tableau algorithms generally work by trying to *construct* models for the given KB.

Numerous implementations of such procedures for the standard DL reasoning tasks have been consolidated in software reasoning engines called *DL reasoners*. Some of these, for example CEL [17], have specialised designs and optimisations for low-complexity DLs, while others such as HermiT [78], are developed for very expressive DLs. More details about reasoning in DLs can be found in the DL Handbook [10] or other provided references.

Finally, we mention that in addition to the standard reasonings tasks discussed earlier in this section, there are various *non-standard* reasoning tasks [113] that have been developed to meet specialised requirements in the ontology engineering setting. We conclude with a brief discussion of some of the more prominent of these.

Classification: A commonly used, informative representation of a KB, is the so-called *subsumption hierarchy* (also known as the *concept hierarchy* or *taxonomy*). It can be defined as the explicit representation of the subsumption relationship between each pair of concept names in the KB, as a directed acyclic graph (DAG).

Example 3 *Let* SM *be an abbreviated concept name representing the "San Marzano" variety of tomato. Consider the following KB:*

$$
\begin{array}{lcl}
\mathsf{NonVegPizza} & \equiv & \mathsf{Pizza} \sqcap \neg\mathsf{VegPizza}, \\
\mathsf{ItalianPizza} & \equiv & \mathsf{Pizza} \sqcap \forall\mathsf{topping}.(\exists\mathsf{origin}.\mathsf{Italy}), \\
\mathsf{VegPizza} & \equiv & \mathsf{Pizza} \sqcap \neg\exists\mathsf{topping}.\mathsf{MeatTopping}, \\
\mathsf{Pizza1} & \sqsubseteq & \mathsf{Pizza} \sqcap \forall\mathsf{topping}.(\mathsf{Avocado} \sqcup \mathsf{Ricotta} \sqcup \mathsf{SM}), \\
\mathsf{Pizza1} & \sqsubseteq & \exists\mathsf{topping}.\mathsf{Avocado} \sqcap \exists\mathsf{topping}.\mathsf{Ricotta} \sqcap \exists\mathsf{topping}.\mathsf{SM}, \\
\mathsf{Pizza2} & \sqsubseteq & \mathsf{Pizza} \sqcap \forall\mathsf{topping}.(\mathsf{Mozzarella} \sqcup \mathsf{Prosciutto} \sqcup \mathsf{SM}), \\
\mathsf{Pizza2} & \sqsubseteq & \exists\mathsf{topping}.\mathsf{Mozzarella} \sqcap \exists\mathsf{topping}.\mathsf{Prosciutto} \sqcap \exists\mathsf{topping}.\mathsf{SM}, \\
\mathsf{Pizza3} & \sqsubseteq & \mathsf{Pizza} \sqcap \forall\mathsf{topping}.(\mathsf{Ricotta} \sqcup \mathsf{SM}), \\
\mathsf{Pizza3} & \sqsubseteq & \exists\mathsf{topping}.\mathsf{Ricotta} \sqcap \exists\mathsf{topping}.\mathsf{SM}, \\
\mathsf{Prosciutto} & \sqsubseteq & \mathsf{MeatTopping} \sqcap \exists\mathsf{origin}.\mathsf{Italy}, \\
\mathsf{Avocado} & \sqsubseteq & \mathsf{VegTopping}, \\
\mathsf{SM} & \sqsubseteq & \mathsf{VegTopping} \sqcap \exists\mathsf{origin}.\mathsf{Italy}, \\
\mathsf{Ricotta} & \sqsubseteq & \mathsf{VegTopping} \sqcap \exists\mathsf{origin}.\mathsf{Italy}, \\
\mathsf{Mozzarella} & \sqsubseteq & \mathsf{VegTopping} \sqcap \exists\mathsf{origin}.\mathsf{Italy}, \\
\mathsf{VegTopping} & \sqsubseteq & \neg\mathsf{MeatTopping}
\end{array}
$$

The task of computing the concept hierarchy is termed *classification*. Classification reveals the subsumption relationship between concept names to a knowledge engineer, that might otherwise be non-obvious in its original axiomatic representation. Applied to Example 3 on Page 34, we illustrate the resulting taxonomy in Figure 2.1.

In the worst case, given the n concept names in a KB, it would take $n^2 - n$ subsumption entailment tests to compute the hierarchy. However, in prac-

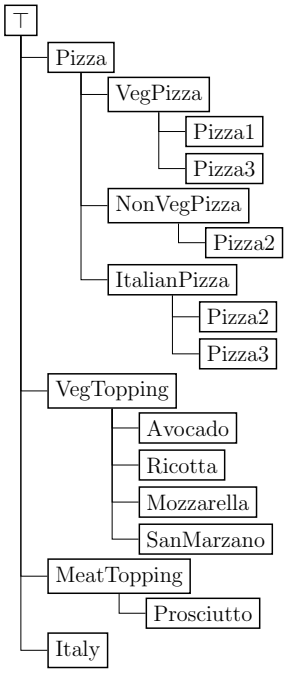


Figure 2.1: Class hierarchy for Example 3.

tice, with the development of an *enhanced traversal* algorithm [13] and subsequent optimisations [179, 77], in addition to novel techniques [76], one can vastly reduce this number. Current implementations can classify a KB on demand even if its size is in the order of hundreds of thousands of concept names [115, 106].

Least Common Subsumer: Given a KB $\mathcal{K}$ and a set of concepts $\mathscr{C} = \{C_1, ..., C_n\}$, a concept C is a *common subsumer* for $\mathscr{C}$ w.r.t. $\mathcal{K}$ if $\mathcal{K} \models C_i \sqsubseteq C$ for $1 \leq i \leq n$. C is a *least* common subsumer (LCS) for $\mathscr{C}$ w.r.t. $\mathcal{K}$ if there is no common subsumer C' for $\mathscr{C}$ w.r.t. $\mathcal{K}$ s.t. $\mathcal{K} \models C' \sqsubseteq C$ and $\mathcal{K} \not\models C \sqsubseteq C'$ [55, 56, 14, 7, 21].

In Example 3 on Page 34, a LCS for $\{\mathsf{Pizza2}, \mathsf{Pizza3}\}$ is $\mathsf{ItalianPizza}$, although this is quite a simplistic example because we restrict ourselves to concept names. However, in general we can compute the LCS for general concept expressions and the LCS itself may also be a concept expression.

Furthermore, in general, there need not exist an LCS for a given set of concept descriptions. Thus the problem is a non-trivial one.

LCS computation has various applications in *vivification* [120, 34] and "bottom-up" ontology development, by enriching the KB with the LCS for a chosen set of terms and its axiomatic relationship with each of these terms [15]. That is, explicitly representing in the KB, the most specific concept descriptions which capture the common properties of the given terms.

Unification: There are various scenarios in which independent KBs make use of the "same" terms. If some of these KBs are integrated, it is useful to avoid redundancy in the form of term duplication.

Example 4 *Consider the following concept expressions:*

1. $\mathsf{Mother} \sqcap \forall \mathsf{child}.\mathsf{Mother}$,
2. $\mathsf{Parent} \sqcap \neg\mathsf{Male} \sqcap \forall \mathsf{child}.(\mathsf{Parent} \sqcap \neg\mathsf{Male})$

Expressions 1 and 2 of Example 4 on Page 36 are meant to represent the same concept - individuals that are mothers of only daughters who are also mothers. However, these expressions are not *logically* equivalent (having the same models), making it non-trivial to automatically determine such.

Unification [18] is the inference problem: given two concepts C, D, finding a substitution $\mathscr{T}$ such that $\mathscr{T}(C) \equiv \mathscr{T}(D)$, where $\mathscr{T}$ is a function that maps some of the atomic concepts in C, D to an arbitrary DL concept expression.

It is easy to see that for Example 4 on Page 36, if we choose a $\mathscr{T}$ which maps the atomic concept Mother in each expression to the expression $\mathsf{Parent} \sqcap \neg\mathsf{Male}$, then this substitution *unifies* Expressions 1 and 2.

The idea with unification is to offer a service highlighting distinct concept expressions in the KB which could possibly be intended to capture the same knowledge. After integration of knowledge from different sources (independent KBs), the likelihood of distinct concept expressions intending to capture the same knowledge increases. KB integration is an example of a suitable setting in which to apply unification in order to avoid redundancy.

Explanation: In KBs with many axioms it can sometimes be difficult to understand the specific reasons *why* a particular entailment holds. This problem has been a source of frustration for ontology engineers since the advent of software ontology editors such as Protégé[1], TopBraid Composer[2], OntoStudio[3] and the NeOnToolkit[4].

Justification-based explanation [94] has been the most successfully applied solution to this problem. The approach presents to the user the minimal (w.r.t. set inclusion) subsets of the KB that entail the axiom in question (called *justifications*), as the concise reasons for the entailment. Justifications are sometimes called *minAs* [19] and *MUPSes* [174] in the literature. There are three categories of algorithm (coined by Parsia et al. [151]) for computing justifications, namely, *blackbox*, *glassbox* and *hybrid* algorithms.

Blackbox algorithms [94, 104] use a DL reasoning implementation purely as an "oracle" to test entailment of the axiom in question. The basic idea is to devise clever strategies to shrink (remove axioms from), and extend (add axioms to) the ontology until a minimal subset that entails the relevant axiom is reached. As mentioned, the reasoner is employed in these algorithms only to periodically test if the entailment still holds in the progressively smaller subsets that are devised.

The main optimisations for these procedures seek to minimise the number of these tests which are the source of greatest complexity in the algorithms. The main advantage of blackbox approaches is their independence of DL reasoning implementations, making them possible to be used in accompaniment with any sound and complete reasoning procedure.

Glassbox approaches [174, 140], in contrast, use sophisticated labelling and tracing techniques in the *tableau construction itself* to identify the minimal information contributing to the entailment of the given axiom. There are also other reasoners (that are not based on tableau procedures) which use similar

[1] protege.stanford.edu

[2] topquadrant.com

[3] semafora-systems.com

[4] neon-toolkit.org

labelling tracing techniques. Since glassbox techniques require modification of the underlying internals of the reasoner in question, they are therefore dependent on the implementation.

Hybrid approaches, as the name suggests, try to combine strategies from both blackbox and glassbox methods [104, 188, 112].

Repair: The area of repair in non-standard reasoning for DLs, asks the question: we have a set of undesired or unintended entailments in our KB, what are the "minimal" or "best" (for some definition thereof) modifications we could make to the KB in order to no longer have these entailments?

Syntactic approaches examine the sentences in the KB in order to answer this question. For example, whereas justifications represent the *minimal* subsets of the KB that entail some axiom(s), repair is concerned with the *maximal* subsets of the KB that do *not* entail the unintended axioms. Minimality of modification to the original ontology is thus a priority in this area which is related closely with Belief Revision [145, 68].

In addition to syntactic approaches there are also *semantic* approaches [118, 156] to find minimal change to the knowledge in the KB. The latter approaches define minimal change w.r.t. the models for the KB. The starting point is usually defining a measure of *distance* or *difference* between two arbitrary models and trying to minimise this distance measure between the models for the original KB and the repaired KB which no longer entails the relevant knowledge.

2.1.3 Trade-off between Expressivity and Complexity

As mentioned earlier, the family of DLs vary according to the set of logical features they provide, and by extension, the worst case computational complexity of their decision procedures. The spectrum of DLs can, in a broad sense, be categorised into three groups: low, medium and high-expressivity or complexity DLs. Such a categorisation can be subjective since it is highly dependent on the spectrum of features required in different applications.

Nevertheless, there is a general consensus in the literature on what constitutes a low, medium or high expressivity DL. In this section, we discuss popular DLs in each category, their relationship to each other in terms of expressivity and complexity and the applications for which they are suitable.

The $\mathcal{EL}$ and DL-Lite families: Some prominent examples of low-complexity or so-called "light-weight" DLs include those in the $\mathcal{EL}$ [9] and DL-Lite families [5]. These formalisms provide relatively few logical features in the pursuit of major reasoning performance, and both families are polynomial-time decidable (tractable) w.r.t. the standard DL reasoning tasks.

The logic $\mathcal{EL}$ is a representative example of this family. Its concept language merely consists of the top concept ($\top$), DL conjunction ($\sqcap$) and existential quantifiers on roles ($\exists$).

An obvious positive of these restricted languages is their tractability which happens to lead to extremely good practical performance [16, 106]. The question is if one is able to capture meaningful and useful knowledge with such a restricted logic.

There is evidence which suggests that these logics have useful application with a case in point being that $\mathcal{EL}$ is the underlying language used for representing the large-scale biomedical ontology SNOMED [57, 184, 64].

In fact, $\mathcal{EL}$ underscores numerous other prominent ontologies in the domain of life sciences, including the Gene Ontology (GO) [6] and restricted versions of Galen [166, 159]. The necessity conditions of a drug overdose, for example, is captured in SNOMED by the $\mathcal{EL}$ axiom $\mathsf{Drug\_overdose} \sqsubseteq \mathsf{Drug\text{-}related\_disorder} \sqcap \exists \mathsf{rolegroup}.(\exists \mathsf{causative\_agent}.\ \mathsf{Drug\_or\_medicament})$.

It turned out that, while $\mathcal{EL}$ is sufficient for representing some knowledge in biomedical applications, it is insufficient for representing certain types of knowledge about roles. One may wish to capture that if an anatomical organ a is *part of* another anatomical organ b, then it a is also *contained in* b. $\mathcal{ELH}$ extends $\mathcal{EL}$ with simple role inclusions (SRIs), represented by the letter $\mathcal{H}$ for *hierarchy* of roles. SRIs, as an additional type of TBox axiom, capture

subsumption relationships between atomic roles (role names).

We can represent the "part-of" and "contained-in" relationship mentioned earlier by the SRI $\mathsf{partOf} \sqsubseteq \mathsf{containedIn}$. The good news is that this additional feature comes "for free" complexity-wise because $\mathcal{EL}\mathcal{H}$ remains tractable [36]. In medical terminologies, however, it is important to be able to represent so-called *right-identity* rules [150, 183, 100]. For example, in the anatomy of the human body, we know that the femur bone is composed of different parts or regions, namely, the head, neck and shaft. Common sense tells us that a fracture of one of these parts of the femur can be regarded generally as a fracture of the femur as a whole.

Example 5 *Consider the following KB:*

$$\begin{aligned} &\mathsf{FemurFracture} && \equiv && \mathsf{Fracture} \sqcap \exists \mathsf{location.Femur}, \\ &\mathsf{FemurShaft} && \sqsubseteq && \exists \mathsf{partof.Femur}, \\ &\mathsf{Fracture}(\mathsf{fracture1}), \\ &\exists \mathsf{location.FemurShaft}(\mathsf{fracture1}) \end{aligned}$$

It is clear that we cannot conclude $\mathsf{FemurFracture}(\mathsf{fracture1})$ from the KB in Example 5 on Page 40, even though this is an intuitive conclusion to make from a common sense point of view. The problem, of course, is that DLs cannot identify the "common sense" that if a is located in b and b is a part of c then a should be located in c as well, because there is no special interpretation of the location and partof relations in DLs. Nevertheless, there are possible modelling solutions to this that do not require additional expressivity.

One possibility is to add FemurShaft as a disjunct to the first axiom in Example 5 on Page 40. This would give us the axiom $\mathsf{FemurFracture} \equiv \mathsf{Fracture} \sqcap \exists \mathsf{location.}(\mathsf{Femur} \sqcup \mathsf{FemurShaft})$. This is not a particularly elegant solution, especially considering that one needs to add such a disjunct for each part of the femur in order to cater for different possible sub-locations of femur fractures. Of course, this solution would take us beyond the realm of $\mathcal{EL}$ (disjunction is disallowed in $\mathcal{EL}$).

A more elegant solution could be changing the first axiom to $\mathsf{FemurFracture} \equiv \mathsf{Fracture} \sqcap \exists \mathsf{location}.(\mathsf{Femur} \sqcup \exists \mathsf{partof}.\mathsf{Femur})$. This is a respectable solution but we have to repeat this pattern of modelling in order to cater for fractures of other bones (e.g. the tibia etc.) if we so wish. □

Because of the frequency, in medical domains, of the kind of modelling problem in Example 5, an additional DL feature was devised called *role chaining* [100] or *role composition*. The role subsumption $\mathsf{P} \circ \mathsf{Q} \sqsubseteq \mathsf{R}$ (read as the composition of P and Q is subsumed by R) captures the constraint that if a is P-related to b, which in turn is Q-related to c, then a is R-related to c.

Role chaining is not restricted to two role names as in our discussion. An arbitrary number of role names (on the left hand side of the inclusion) are allowed. Role chaining allows us to add the role inclusion (RI) $\mathsf{location} \circ \mathsf{partof} \sqsubseteq \mathsf{location}$ to the KB in Example 5 on Page 40 and immediately obtain the desired conclusion $\mathsf{FemurFracture}(\mathsf{fracture1})$.

$\mathcal{ELH}$ together with the capability of role chaining (on the left hand side of role subsumptions) results in the DL $\mathcal{EL}^+$ [17] which is supported by the CEL implementation. It is worth mentioning that SNOMED makes use of role chaining in a few of its axioms which technically makes it an $\mathcal{EL}^+$ ontology.

$\mathcal{EL}^{++}$ [9] is a further enhancement to $\mathcal{EL}^+$, adding the bottom concept ($\bot$), nominals (singleton concepts) and concrete domains [127] (e.g. references to numbers and strings). $\mathcal{EL}^+$ and $\mathcal{EL}^{++}$ remain the most prominent variants of $\mathcal{EL}$ that are tractable, and the aforementioned variants of $\mathcal{EL}$ form the logical underpinning of the OWL 2 EL profile.

The DL-Lite family of DLs [5] are a group of tractable logics that have been designed with very specific application scenarios in mind. The primary applications lie in ontology-based data access (OBDA) [61, 92, 152], as alternative formalisms for concept modelling (thereby providing automated reasoning services for such models) [47, 35, 67, 138], information and data integration [46, 80, 147, 139]. OBDA has been a prominent topic in DLs in recent times. The basic idea is that we have a physical data source (usu-

ally a relational database) and auxiliary knowledge about the application domain pertaining to the data (in the form of a DL TBox). The task is to answer queries about instance data while taking into account knowledge in the TBox.

A simple example can be given concerning data about patients in a medical facility. Suppose we have a database which purely stores information about the systolic pressure readings of patients and their relatives[5]. A doctor might obviously have knowledge about what systolic pressure readings constitute a case of hypertension. Furthermore, she may know that a patient who has a family history of hypertension is at risk of myocardial infarction (getting a heart attack).

This knowledge is not contained in the database itself, but she can encode it in a DL TBox without modifying the database, and then ask queries about patients using the terminology in the TBox. For example, she can query the database to find the patients at risk of getting a myocardial infarction.

Because of the specialised requirements of such applications (queries have to be performed over very large databases relative to the number of facts in the TBox), certain restrictions have to placed on the language of the TBox to make reasoning efficient in such a setting. The resulting DL-Lite family disallows all disjunction and qualifications on existential role restrictions ($\exists R.C$ is not allowed where $C \neq \top$) and have been proposed as reasonable candidates for application in tasks such as OBDA.

The significance of the DL-Lite family is highlighted by the fact that it forms the basis of the OWL 2 QL profile (a targeted set of sub-languages of the OWL 2 standard). The QL in OWL 2 QL stands for Query Language, alluding to the application of query answering over large data sources. For detailed information about the DL-Lite family and applications, the reader should consult the provided references.

[5]Note that this scenario is similar to that of Example 2 on Page 32, although the knowledge expressed in that example cannot be represented in DL-Lite. An alternative formulation of the knowledge would thus be required.

Very Expressive DLs: We briefly discuss some of the more expressive DLs that form the basis of the OWL 2 DL (The full language of OWL 2). In Section 2.1.1, we have presented the moderate complexity DL $\mathcal{ALC}$ which is already intractable (w.r.t. the problem of checking concept satisfiability). TBox entailment in $\mathcal{ALC}$ is an EXPTIME-COMPLETE problem [172].

Even though $\mathcal{ALC}$ is sufficient for accurate and relatively precise representation of knowledge from many application domains, there are various types of knowledge which it cannot capture.

For example, in $\mathcal{ALC}$ we can capture that dogs have *at least one* leg ($\mathsf{Dog} \sqsubseteq \exists \mathsf{hasLeg}.\top$), but we cannot express that a dog has *exactly* four legs ($\mathsf{Dog} \sqsubseteq = 4\ \mathsf{hasLeg}.\top$) or that land mammals have *at least two* legs ($\mathsf{Mammal} \sqcap \forall \mathsf{hasHabitat.Land} \sqsubseteq \geq 2\ \mathsf{hasLeg}.\top$). Such features are known as *number restrictions* or *cardinality restrictions* on roles.

Recall that SRIs cannot be expressed in $\mathcal{ALC}$ either. Another useful feature pertaining to roles, absent from $\mathcal{ALC}$, is called *inverse roles*. Inverse roles allow one to state for example that the $\mathsf{isIngredientOf}$ role is the inverse of $\mathsf{hasIngredient}$ ($\mathsf{hasIngredient} \equiv \mathsf{isIngredientOf}^{-}$).

Thus by stating that $\mathsf{hasIngredient}$ ($salad\_instance1$, $lettuce\_instance1$) one can immediately infer $\mathsf{isIngredientOf}$ ($lettuce\_instance1$, $salad\_instance1$).

If we extend $\mathcal{ALC}$ with all the features mentioned above (and allowing transitivity of roles), the resulting logic is called $\mathcal{SHIN}$. If, additionally, we allow role fillers for cardinality restricted roles to be concepts other than $\top$ (i.e., we allow number restrictions to be *qualified*), then we have the DL $\mathcal{SHIQ}$. Therefore, we can express that dogs have exactly two *hind* legs ($\mathsf{Dog} \sqsubseteq = 2\mathsf{hasLeg.HindLeg}$) in $\mathcal{SHIQ}$.

Interestingly, in $\mathcal{SHIN}$ (and $\mathcal{SHIQ}$), we are allowed to state expressions that require an interpretation domain with an infinite number of elements in order to construct models for them [103, Section 3.4]. When this is true of a logic, then we say that it does not possess the *finite model property (FMP)*. Extending $\mathcal{SHIQ}$ with the ability to express singleton concepts called *nom-*

inals (we can state $\{\mathsf{usain\_bolt}\} \sqsubseteq \mathsf{Sprinter}$), we obtain the DL $\mathcal{SHOIQ}$.

$\mathcal{SHOIQ}$, together with various properties that one can enforce on roles [99], results in the DL $\mathcal{SROIQ}$ which is the logical underpinning of (and the most expressive DL supported by) the OWL 2 standard.

2.2 The Web Ontology Language (OWL)

The Web Ontology Language (specifically its latest version OWL 2) is a W3C recommendation for representing ontologies on the Semantic Web [25]. OWL is logically underpinned by DLs and provides additional non-logical features such as annotations for storing meta-data about ontologies. Ontologies expressed with OWL are mostly serialised in a variety of concrete XML-based syntaxes, such as RDF/XML, Manchester OWL and OWL/XML.

Because of its correspondence with DLs, reasoning with OWL ontologies is accomplished by exploiting the common reasoning tasks of the underlying DL used to express the ontology. While there are a variety of concrete syntaxes mentioned above for OWL, there is just one *formal* syntax. However, there are two semantics: direct semantics and RDF-based semantics.

The language defined by the full OWL 2 syntax, together with direct semantics is called OWL 2 DL. OWL 2 defines three *profiles* or sub-languages targeted for different application settings. The categorisation is made according to the trade-off between expressivity and computational complexity. The three profiles are OWL 2 EL, OWL 2 QL, OWL 2 RL These languages have been discussed in Section 2.1.3 save for OWL 2 RL.

OWL 2 RL is a profile designed to restrict the expressivity of OWL 2 to a language that can be implemented using rule-based technologies such as Description Logic Programs (DLPs) [87]. OWL 2 DL remains the language of interest for our purposes of possible integration of defeasible features.

2.3 Incomplete Knowledge

Knowledge is prone to imperfections. The primary method by which humans obtain knowledge is through the senses, which are themselves imperfect and limited. The limited nature of the senses necessarily leads to the acquisition of knowledge that is *incomplete* in the broadest sense of the word.

The seemingly endless refinement and evolution of this incomplete knowledge indicates that it is subject to change with time. Perfect knowledge, in contrast, has to be unchanging and *absolute*, without need of qualification. Unfortunately, such knowledge still eludes humanity at large.

Yet, despite the inherent incompleteness of our knowledge, we humans have to infer things from it to accomplish various tasks, and in keeping with this, our natural languages are enriched with vocabulary to express incompleteness. We say things like "lawyers are *usually* dishonest", "It is *likely* that it will rain tomorrow", "James is *roughly* 50 years old" etc., where the emphasised words in these phrases allude to a lack of complete information.

Although classical logics cater for *some* forms of incompleteness in knowledge, they are much more limited than natural languages in this regard. One particular example of incomplete knowledge that classical logics can represent is *disjunction*. For example, if I flip a coin, even if I don't know what the outcome of this flip will be, I do know that it will either be heads or tails.

With regards to us humans, the incompleteness of our knowledge invariably leads us to employ "guesswork" when explicitly representing knowledge. Some may use all relevant information at their disposal and apply this information sensibly to inform their guesswork, others may not.

It is the symptoms of this guesswork which indicate that we do not have access to complete knowledge. The three *main* symptoms in KR are: *defeasibility*, *uncertainty* and *subjectivity or relativity*. We shall discuss these and mention which of these we are specifically interested in for this thesis.

2.3.1 Defeasibility

A defeasible statement is one that is potentially fallible or can be refuted upon the discovery of more information.

Students *normally* don't pay taxes, sushi *usually* contains fish, birds *typically* can fly are all sentences are all punctuated by terms which express *generalisations* that are inapplicable to *exceptions*. For example, employed students, vegetarian sushi and ostriches or penguins represent exceptions to the mentioned defeasible statements. Therefore, it is sometimes useful to ignore exceptions where possible and make generalised statements in order to reason and derive *plausible* inferences from such knowledge.

However, in classical logics, there is no native machinery that allows us to represent generalisations that admit exceptions.

2.3.2 Uncertainty

Particular types of knowledge, linked to statistical information and probabilities, are expressed because of underlying uncertainty. Sentences like: "1 in 4 people will contract cancer at some point in their lives" and "there is a 60% chance that David has diabetes" represent empirically determined likelihoods of events happening. The computation and representation of these likelihoods (or probabilities) are only necessary because we do not know, a priori, all individuals who will contract cancer (or respectively, whether David has diabetes or not).

Fortunately, this kind of representation is supported in many classical probabilistic logics [146]. Classical logics that do not natively support probabilities also cater for certain forms of uncertainty. For example, if I know that *john* is either a soccer player or a tennis player, it represents a kind of uncertainty about which category *john* belongs to. Of course, this kind of uncertainty can be expressed using disjunction which is native to many classical logics including DLs.

2.3.3 Subjectivity or Relativity

John is "tall", Suzie is "young" and this hotel is "cheap" are examples of sentences which contain terms that are *subjective* or *relative* because there are no known definitions of these terms that are absolute.Therefore, a relative definition is adopted in order to attach meaning to these statements.

There are various extensions of classical logic (often called *fuzzy logics* [187]) which enable one to specify subjective meanings for these terms. In some cases, subjective terms (especially those that can be interpreted numerically) can be defined in classical logics. For example, one can use datatypes [144] in OWL to specify a term "adultAge" and assign it a value, e.g. 18, which represents an objective minimum age for an adult.

This thesis is particularly concerned with addressing the specific type of defeasible knowledge (Section 2.3.1) in which we would like to make generalised statements that permit exceptions. For example we would like to express that students *generally* don't pay taxes which is not falsified even upon discovering that students who are employed in certain contexts are obliged to pay taxes. We now analyse the state-of-the-art of various approaches in the literature which aim to represent and reason with such knowledge.

2.4 Circumscription

Circumscription is a class of nonmonotonic logics proposed by John McCarthy in 1980 [136]. The basic aim is to be able to represent, in KR formalisms, the presumption that something is "normal" and behaves as expected *unless* otherwise specified.

2.4.1 Basic Circumscription

Consider the following version of the checkerboard example from Raymond Reiter [164]:

Example 6 *Suppose we have a black and white checkerboard and we toss a coin over the checkerboard. We have to model the problem of some agent predicting where the coin will land.*

Essentially, there are three possibilities in Example 6: either the coin lands on a white square, black square or partially over both. In classical logics it is easy to represent this knowledge using disjunction: $(\forall x, y)$ $(\mathsf{hasCoinTossResult}(x, y) \implies \mathsf{black}(y) \vee \mathsf{white}(y))$. The problem is how classical logics interpret *exceptions* to this.

Suppose the coin lands on the floor, on the table next to the board, on the moon, or anywhere else outside the boundaries of the board. The representation becomes: $(\exists x, y)(\mathsf{hasCoinTossResult}(\mathsf{x}, \mathsf{y}) \wedge \neg\mathsf{black}(\mathsf{y}) \wedge \neg\mathsf{white}(\mathsf{y}))$. No models for this accumulated knowledge exist.

The cases of the coin landing outside the boundaries of the checkerboard, while possible, are *exceptional* and not in the "spirit" of solving the problem at hand. We would not like the agent to consider them unless it really has to. One way to eliminate these situations from consideration is to add them as "qualifications" to the agent's knowledge about the system.

That is, we can rephrase the problem as follows: if the coin is tossed over the checkerboard *and* it does not land on the floor *and* it does not land on the table *and* it does not land on the moon etc., then where does it land? The obvious drawback of this solution is that the possible qualifications can be endless or not known a priori. □

Thus we require a formalism which (i) permits exceptions (unlike classical logics), since it is possible for them to occur, and (ii) does not require one to explicitly list such exceptions. Circumscription models the tentative assumption that the information we have is *complete* and any additional information that we encounter, which conflicts with what we know, can be immediately identified as an *abnormal* case.

For Example 6 on Page 48, the consideration that the checkerboard is the only place where the coin can land is our assumption of "complete" in-

formation. The treatment of circumstances in which the coin lands outside the checkerboard, as abnormal cases, is the heart of the circumscriptive technique.

The pioneers of the circumscriptive technique were McCarthy and Lifschitz [136, 137, 121, 124], whose expositions were primarily in the setting of first-order logic. In this thesis we are dealing exclusively with DLs, therefore we give here the state-of-the-art of circumscription applied to DLs [32].

The main approach for introducing circumscription into DLs is what is called *Predicate Circumscription*. The basic idea of the approach is to add concept and role names (collectively, these are sometimes referred to as abnormality predicates) to the vocabulary of the KB. The job of the additional concept and/or role names are to account for the abnormal or exceptional objects in the models.

Given abnormality predicates, it is only necessary to consider those models of the KB in which the extension of these predicates are *minimal* (w.r.t. set inclusion). Such models are generally known as *minimal models*.

The intuition behind this necessity is that we should consider the most "normal" situations possible when making inferences, in which there are as few exceptional cases as possible (further motivating examples are given by Lifschitz [124] and McCarthy [137]). Indeed it is not hard to notice, then, that circumscription will favour models in which the extensions of the abnormality predicates are empty (favouring situations in which there are no exceptions). Non-empty extensions will thus only be considered if there is evidence (information in the KB) which enforces them to be such.

In order to identify minimal models, a preference relation, $<_{\mathcal{M}}$, is defined over all models. Suppose N_c, N_r and N_i are respectively the set of concept names, role names and individual names representing the vocabulary of our KB. In addition, suppose that $\mathcal{M} \subseteq \mathsf{N}_c \cup \mathsf{N}_r$ denotes the set of predicates we wish to minimize. An interpretation $\mathcal{I}$ of the KB is "preferred" over an interpretation $\mathcal{J}$ (written $\mathcal{I} <_{\mathcal{M}} \mathcal{J}$) [32] if:

1. $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$ and for all $a \in \mathsf{N}_i$, $a^{\mathcal{I}} = a^{\mathcal{J}}$;

2. for all $p \in \mathcal{M}$, $p^{\mathcal{I}} \subseteq p^{\mathcal{J}}$;

3. there is a $p \in \mathcal{M}$ such that $p^{\mathcal{I}} \subset p^{\mathcal{J}}$;

The first condition ensures that both models refer to the same domain, the second and third are the main conditions requiring consideration of only those models in which the abnormality predicates are minimal w.r.t. set inclusion.

Specifically, the second condition makes sure that the more preferred model, $\mathcal{I}$, does not interpret any minimised predicate to have *more* members than the less preferred model, $\mathcal{J}$.

The third requires *at least one* minimised predicate to have *fewer* members in the more preferred model, than in the less preferred one.

In some versions of circumscription, a fourth condition is required to define the preference relation: (4) for all $p \in (\mathsf{N}_c \cup \mathsf{N}_r) \backslash \mathcal{M}$, $p^{\mathcal{I}} = p^{\mathcal{J}}$; This condition enforces that the two models should interpret all other symbols (non-minimised) in exactly the same way, in order to be comparable via the preference relation.

This is also called *fixing* the extension of the selected predicates. The only implication of including or excluding this condition is obviously the impact on the conclusions one is able to draw, since we are generally looking at a different set of models in each case.

If the condition is excluded, we are essentially allowing the comparison of models (via $<_{\mathcal{M}}$) which *differ* in their extensions of the non-minimised predicates. In general, when we allow this (the non-minimised predicates to *vary*) then it results in the derivation of *more* conclusions (fewer minimal models) and when we fix these predicates by including condition (4), then we derive fewer conclusions (more minimal models) [32].

The decision of which predicates to minimise, fix and allow to vary is called a *circumscription pattern*. It is the circumscription pattern that determines the models we are looking at to decide entailment. Note that the non-minimised predicates do not have to be exclusively fixed or exclusively varied (simulated by the inclusion or exclusion of condition (4) above).

The user is given freedom to separate the non-minimised predicates into those he/she would like to fix and those he/she would like to allow to vary, resulting in a refined version of condition (4).

We give an example illustrating the impact of the circumscription pattern on the entailment relation:

Example 7 *Consider the following KB:*

1. $\mathsf{Student} \sqsubseteq \neg\exists\mathsf{receives}.\mathsf{TaxInvoice} \sqcup \mathsf{Ab}_{\mathsf{Student}}$
2. $\mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \sqcap \exists\mathsf{receives}.\mathsf{TaxInvoice}$

Axiom 1 of Example 7 on Page 51 intuitively says that students normally do not pay taxes (either that or they are abnormal students). Axiom 2 says that employed students are students that are compelled to pay taxes. Intuitively speaking, a sensible inference of this knowledge should be that employed students are abnormal students ($\mathsf{EmployedStudent} \sqsubseteq \mathsf{Ab}_{\mathsf{Student}}$) and thus it should still be rational to additionally infer that employed students pay taxes (Axiom 2), even though students *normally* don't pay taxes (Axiom 1).

If we choose the circumscription pattern that minimises $\mathsf{Ab}_{\mathsf{Student}}$ and fixes all other predicates in the KB, then we indeed obtain the desired inferences mentioned above. In addition, we also obtain that abnormal students are exactly those students who pay taxes ($\mathsf{Ab}_{\mathsf{Student}} \equiv \mathsf{Student} \sqcap \exists\mathsf{receives}.\mathsf{TaxInvoice}$).

This is all fine and well, but one may find it sensible to also assume that exceptions to Axiom 1 are highly unlikely to the point of treating the explicitly stated exceptions as the *only* exceptions. I.e., we may want to conclude that $\models \mathsf{Ab}_{\mathsf{Student}} \equiv \mathsf{EmployedStudent}$.

Since Example 7 on Page 51 has only one explicitly represented exception - employed students - it should be reasonable to conclude that employed students are the *only* abnormal students (leading to $\mathsf{EmployedStudent} \equiv \mathsf{Ab}_{\mathsf{Student}}$) until further notice. However, using the current circumscription pattern does not yield this entailment. The following counter-model, $\mathcal{I}$,

proves this: $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathsf{Student}^{\mathcal{I}}, \mathsf{Ab}^{\mathcal{I}}_{\mathsf{Student}} = \{a\}$, $\mathsf{TaxInvoice}^{\mathcal{I}} = \{b\}$, $\mathsf{receives}^{\mathcal{I}} = \{(a, b)\}$, $\mathsf{EmployedStudent}^{\mathcal{I}} = \emptyset$.

It is easy to see that this model minimises the extension of $\mathsf{Ab}_{\mathsf{Student}}$, that is, there is no model in which the extension of $\mathsf{Ab}_{\mathsf{Student}}$ is empty (smaller than in $\mathcal{I}$) while keeping the extension of all the other predicates the same as in $\mathcal{I}$. But $\mathcal{I}$ does not satisfy the axiom $\mathsf{Ab}_{\mathsf{Student}} \sqsubseteq \mathsf{EmployedStudent}$ and therefore cannot satisfy $\mathsf{EmployedStudent} \equiv \mathsf{Ab}_{\mathsf{Student}}$.

A solution is to change the circumscription pattern. We find that allowing receives and TaxInvoice to vary (fixing the remainder of the predicates) gives us the additional inference we desire - $\mathsf{EmployedStudent} \equiv \mathsf{Ab}_{\mathsf{Student}}$. We can see that, using this pattern, the interpretation $\mathcal{I}$ mentioned earlier is no longer a minimal model for our KB. One possible minimal model is $\mathcal{J}$:

$\Delta^{\mathcal{J}} = \{a\}$, $\mathsf{Student}^{\mathcal{J}} = \{a\}$, $\mathsf{EmployedStudent}^{\mathcal{J}} = \mathsf{Ab}^{\mathcal{J}}_{\mathsf{Student}} = \mathsf{TaxInvoice}^{\mathcal{J}} = \mathsf{receives}^{\mathcal{J}} = \emptyset$ and it is obvious that $\mathcal{J} \Vdash \mathsf{EmployedStudent} \equiv \mathsf{Ab}_{\mathsf{Student}}$.

Of course, it may also be sensible in some applications to assume that exceptional things (e.g. employed students) do not exist *unless* it is explicitly stated that they do. Indeed this is in keeping with the mandate of the circumscription discussed in Example 6 on Page 48. This would mean that we should be able to conclude $\mathsf{EmployedStudent} \equiv \mathsf{Ab}_{\mathsf{Student}} \equiv \bot$ from Example 7 on Page 51.

However, we can only achieve this by also allowing EmployedStudent to vary. Doing so, and then introducing an employed student into the mix by adding the assertion $\mathsf{EmployedStudent}(\mathsf{john})$, results in the intuitive conclusion $\mathsf{EmployedStudent} \equiv \mathsf{Ab}_{\mathsf{Student}} \equiv \{\mathsf{john}\}$. For more information about this latter presumption in circumscription see Section 2.4.3. □

2.4.2 Prioritised Circumscription

Up to now, we have only considered the case of introducing a single abnormality predicate into the KB. Of course, circumscription allows for an arbitrary number of abnormality predicates to be introduced and minimised

during reasoning. Suppose we introduce another abnormality predicate into our knowledge, we obtain the following KB:

Example 8 *Consider the following KB:*

1. $\mathsf{Student} \sqsubseteq \neg\exists\mathsf{receives}.\mathsf{TaxInvoice} \sqcup \mathsf{Ab}_{\mathsf{Student}}$
2. $\mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \sqcap \exists\mathsf{receives}.\mathsf{TaxInvoice} \sqcup \mathsf{Ab}_{\mathsf{EmployedStudent}}$
3. $\mathsf{EmployedStudentParent} \sqsubseteq \mathsf{EmployedStudent} \sqcap \neg(\exists\mathsf{receives}.\mathsf{TaxInvoice})$

Minimising $\mathsf{Ab}_{\mathsf{Student}}$ and $\mathsf{Ab}_{\mathsf{EmployedStudent}}$, allowing $\mathsf{receives}$ and $\mathsf{TaxInvoice}$ to vary and fixing the remainder of the predicates in Example 8 on Page 53, has the consequence that we can derive neither of the axioms $\mathsf{EmployedStudent} \sqsubseteq \exists\mathsf{receives}.\mathsf{TaxInvoice}$ nor $\mathsf{EmployedStudent} \sqsubseteq \neg(\exists\mathsf{receives}.\mathsf{TaxInvoice})$. Essentially, this is because circumscription cannot decisively prefer which abnormality predicate ($\mathsf{Ab}_{\mathsf{Student}}$ or $\mathsf{Ab}_{\mathsf{EmployedStudent}}$) to apply to a given object.

That is, the models in which $\mathsf{Ab}_{\mathsf{Student}}$ and $\mathsf{Ab}_{\mathsf{EmployedStudent}}$ are minimised, are mutually exclusive. If there are no situations in which *both* predicates are minimal w.r.t. all the models, then one has to "prefer" one situation over the other in order to derive one of the conclusions mentioned above.

The question is how this preference is determined among an arbitrary number of abnormality predicates. In the case of Example 8 on Page 53, we only have two minimised predicates and it is argued that the more intuitive conclusion to draw would be $\mathsf{EmployedStudent} \sqsubseteq \exists\mathsf{receives}.\mathsf{TaxInvoice}$ (i.e., preferring to minimise $\mathsf{Ab}_{\mathsf{EmployedStudent}}$ over $\mathsf{Ab}_{\mathsf{Student}}$).

Intuitively speaking, this choice simulates the reasoning paradigm that: given a student who is employed, one would prefer to assign to the student the properties in the KB which are most specifically applicable to *employed* students (Axiom 2), and therefore assume that this student is not an abnormal employed student (minimising $\mathsf{Ab}_{\mathsf{EmployedStudent}}$).

This would be in contrast to preferring to assume that the student is not abnormal (minimising $\mathsf{Ab}_{\mathsf{Student}}$) and being able to assign attributes of a general student (Axiom 1) to them, but enforcing that the student must be

an abnormal employed student so as not to clash with Axiom 2. □

The specificity approach to prioritisation discussed in Example 8 on Page 53 is quite sensible because it starts at the most specific information that we know about an object and starts to "generalise" (attribute more general properties to the object) and stops when there a conflict or contradiction is reached. In fact, to emulate this behaviour with a general number of abnormality predicates, it has been recommended that the preference relation ($<_{min}$) over the predicates should mirror the classical concept hierarchy of the KB [12].

That is, more generally, if KB $\models \mathsf{X} \sqsubseteq \mathsf{Y}$ then $\mathsf{Ab_X} <_{min} \mathsf{Ab_Y}$ (Minimising $\mathsf{Ab_X}$ should be preferred over minimising $\mathsf{Ab_Y}$). In addition, since the subsumption hierarchy is a partial order on the concept names in the KB, the preferences over minimised predicates is also usually a partial order [32].

Using the above approach models a kind of defeasible inheritance or overriding strategy in the reasoning process. The preference relation over the minimised predicates, if respecting the concept hierarchy of the KB, also simulates an intuitive semantic notion: the more specific the category being considered, the more unlikely it is to have exceptions.

In general, however, the preferences over minimised predicates is left up to the user (ontology developer) in circumscription and becomes an extra component of the circumscription pattern of a KB.

2.4.3 Grounded Circumscription

Grounded circumscription [178] introduces a localised version of the *closed world assumption* (CWA) [160] to circumscription, by only allowing individuals explicitly mentioned in the KB to appear in the extension of minimised predicates. This kind of circumscription is applied in situations where assumption of complete information is appropriate in making inferences. That is, where a kind of CWA is suitable.

Such an assumption is useful in certain applications such as *matchmaking*

in web services [85]. A scenario is also given by the conclusion $\mathsf{EmployedStudent} \equiv \mathsf{Ab}_{\mathsf{Student}} \equiv \{\mathsf{john}\}$ in Example 7 on Page 51.

2.4.4 Complexity Considerations

Circumscription has been studied in the context of first-order logics [136, 137, 121, 124] and propositional logics [65, 45] as far back as 1980, but apart from Brewka [38] and Cadoli, Donini and Schaerf [44], it was only in the last two decades that serious interest arose in tackling DLs.

Bonatti, Wolter and Lutz established initial complexity results in 2006 [32] and 2009 [33] by showing that circumscription in $\mathcal{ALCIO}$ and $\mathcal{ALCQO}$ are decidable (both in $\mathsf{NExp}^{\mathsf{EXP}}$) with some preconditions (binary predicates must be allowed to vary). It is also shown that the complexity decreases significantly (to $\mathsf{NP}^{\mathsf{NEXP}}$) if there are bounds placed on the number of minimised and fixed predicates.

More recently, attention was turned to determine the decidability of circumscription in DLs that do not have the *finite model property*. Bonatti et al. [27] show that DL-Lite$_{\mathcal{F}}$ and $\mathcal{ALCFI}$, even though they do not have this property, are both decidable (when using certain reasonable circumscription patterns).

For less expressive DLs such as $\mathcal{EL}$-variants [9] and the DL-Lite family [5], there have been similar complexity investigations [28, 29, 31]. However, since we are primarily interested in handling exceptions in DLs of $\mathcal{ALC}$ or higher expressivity, we focus on the complexity of circumscription in such cases.

In general, practical implementations and performance results for circumscription in DLs is not well represented in the literature. Grimm, Hitzler, Krisnadhi and Sengupta [86, 178] give tableau algorithms for computing circumscription in variants of $\mathcal{ALC}$. However, implementations and evaluations thereof are not documented in great detail.

In fact, in general, the absence of comprehensive empirical investigations into the practical performance of circumscription for DLs is a major problem.

2.4.5 Discussion

Circumscription, although being one of the more mature proposals for dealing with exceptions in DLs, is not without its limitations. One issue is its inherent limitation in dealing with disjunctive information. Depending on how we model knowledge, we can get counter-intuitive results.

Disjunction: Recall Example 6 on Page 48. Given a coin tossed over a black-and-white checkerboard, circumscription was able to model that the coin should land either on a white block or on a black block on the board and not anywhere else. In DLs, a possible formalisation could be represented by the axiom: $\mathsf{CoinToss} \sqsubseteq (= 1\ \mathsf{landsOn}.\mathsf{BlackSquare}) \sqcup (= 1\ \mathsf{landsOn}.\mathsf{WhiteSquare})$.

If we minimise the role landsOn, we eliminate the possibility of the coin landing on the floor or the moon etc. because the models for these relevant scenarios are not minimal w.r.t. landsOn (w.r.t. set inclusion). But, for the same reason, we also obtain the undesired consequence that models in which the coin lands (partially) on *both* a black and white square are not permitted.

One might ask if a different modelling choice could solve the problem. The following formalisation avoids the problem of interpreting the disjunction exclusively: $\mathsf{CoinTossResult} \sqsubseteq \mathsf{BlackSquare} \sqcup \mathsf{WhiteSquare} \sqcup \mathsf{Ab}_{\mathsf{CoinTossResult}}$, if minimising the predicate $\mathsf{Ab}_{\mathsf{CoinTossResult}}$. But, this formalisation also permits counter-intuitive possibilities like the coin landing partially on a black (or white) square *and* partially on the moon (or any other surface)!

Of course, if one had the expressivity of number restrictions then we could actually formalise the coin toss in DLs without even having to resort to circumscription: $\mathsf{CoinToss} \sqsubseteq (= 1\mathsf{landsOn}.\mathsf{BlackSquare} \sqcup = 1\mathsf{landsOn}.\mathsf{WhiteSquare}) \sqcap \forall \mathsf{landsOn}.(\mathsf{BlackSquare} \sqcup \mathsf{WhiteSquare})$. This formalisation would not permit models in which the coin lands outside the checkerboard bounds and will permit models in which the coin lands on both a black and white square.

Nevertheless, in general, the behaviour of circumscription in the presence of disjunctive information is still a significant limitation. Fortunately, a so-

lution is proposed by Eiter et al. [65] called *theory curbing*. Theory curbing addresses the issue by not only permitting minimal models, but also the *least upper bound* models of these minimal models.

In our coin toss example, even though models in which the coin lands on both a black and white square are not minimal, they are "least upper bound" models for the minimal models such that they interpret the disjunctive information *inclusively*. Therefore such models are also considered in the circumscription. For more detailed information about the approach, the interested reader should consult the provided reference above.

Another drawback to circumscription, from a user perspective, is concerning the circumscription pattern.

Circumscription Pattern: As explained in Sections 2.4.1 to 2.4.3, the circumscriptive pattern is integral to determining the inferences one is able to draw from the given knowledge. In fact, it also affects the computational complexity (and in many cases, decidability) of circumscription. In most cases, the user is entirely responsible for deciding the circumscription pattern. While leaving the pattern up to the user provides flexibility in reasoning, we argue that it can unnecessarily complicate matters.

Even if the circumscription pattern could be partially suggested, fully suggested or automated by the reasoning engine, it is not clear how one could formalise this. That is, to date, a formal methodology for determining reasonable circumscription patterns in practice has not yet been developed. This leads us to other issues with the current complexity and implementation landscape of circumscription in DLs.

Complexity and Implementation: Even though there are various expressive DLs for which circumscription is decidable (see Section 2.4.4), these results are dependent on restrictions to the circumscription pattern. Of course, even though decidability is reached, this is not a guarantee of good performance in practice. Furthermore, in most cases the complexity of circum-

scription is substantially higher than the complexity of standard reasoning in the underlying DL.

To compound matters, even though we know of notable implementations and performance evaluations of circumscription in DLs (by Piero Bonatti and colleagues as well as Pascal Hitzler and colleagues), these implementations and evaluations remain unpublished to a large extent.

2.5 Default Reasoning

In 1980, within the setting of first order logic, Raymond Reiter [161] presented a formalism for encoding so-called *defaults*. Defaults, intuitively speaking, enable the representation of the plausible conditions under which a conclusion (first order sentence) could be entailed by a first order theory (KB).

2.5.1 Basic Default Logic

A *general* default is of the form $\frac{\alpha:\beta}{\gamma}$ where α, β and γ are respectively the *prerequisite*, *justification* and *consequent* of the default. Defaults can be read as "if α holds and it is *consistent* to assume that β holds, then γ plausibly holds (i.e., we can believe γ)".

The phrase "consistent to assume that" is the critical part of the definition of a default and is given a formalisation by Reiter in his work. The following example of a default, $\frac{\mathsf{Student}(x):\neg\mathsf{TaxPayer}(x)}{\neg\mathsf{TaxPayer}(x)}$, encodes that, given a general student x, if it is consistent to assume that x does not pay taxes, then it is plausible to conclude that it doesn't.

Defaults of the form $\frac{\alpha:\beta}{\beta}$, like the above example, are called *normal* defaults. Those that contain free variables in α, β or γ are known as *open* defaults, while those that don't are called *closed* defaults.
Formally, a *default KB* is a pair $\langle \mathcal{W}, \mathcal{D} \rangle$, where $\mathcal{W}$ is a finite set of first order sentences, and $\mathcal{D}$ a finite set of defaults of the form discussed. For each default $\delta = \frac{\alpha:\beta}{\gamma}$ s.t. $\delta \in \mathcal{D}$, Reiter explains that one can conclude γ from the KB (written $\langle \mathcal{W}, \mathcal{D} \rangle \vdash \gamma$) if $\neg\beta$ cannot be deduced from $\mathcal{W}$ together with

the *other* defaults in $\mathcal{D}$. A more formal inductive definition is given in his exposition [161, Theorem 2.1].

Reiter envisaged the primary use-case for his defaults as "meta-rules" that one can apply to a first order KB to compute plausible extensions for it. Indeed it is shown that there could be multiple extensions for a given default KB because defaults can "interact" with each other.

Consider the default theory $\langle \mathcal{W}, \mathcal{D} \rangle$ where $\mathcal{W} = \{\mathsf{EmployedStudent}(x) \implies \mathsf{Student}(x), \mathsf{EmployedStudent}(john)\}$ and $\mathcal{D} = \{\frac{\mathsf{Student}(x):\neg\mathsf{TaxPayer}(x)}{\neg\mathsf{TaxPayer}(x)}, \frac{\mathsf{EmployedStudent}(x):\mathsf{TaxPayer}(x)}{\mathsf{TaxPayer}(x)}\}$.

There are two extensions of $\langle \mathcal{W}, \mathcal{D} \rangle$ that are incompatible. In one, we deduce that $john$ is a student and apply the first default deriving that $john$ is not a tax payer - the formula $\neg\mathsf{TaxPayer}(john)$, and in the other, we notice that $john$ is an employed student and apply the second default deriving that $john$ is a tax payer - $\mathsf{TaxPayer}(john)$.

Reiter showed that the case of multiple extensions implies that there are clashes or undesired interaction between defaults [161, Corollary 3.4]. Furthermore, he showed that for general defaults there is a possibility of a default theory to have no extension. This result led him to restrict his view of practically applicable defaults to normal defaults. Indeed most, but not all [165], naturally occurring examples of defaults are normal ones.

It is important to note that Reiter intended his formalism for deriving *plausible* beliefs. That is, for a sentence to be logically deducible in default logic, there need only be at least one extension in which it appears - the so-called *credulous* approach. This is in opposition to the *skeptical* approach of other nonmonotonic logics which require the formula to be in *all* extensions.

Applied to the student and employed student example above, default logic states that $john$ being a tax payer is a plausible inference to make from the KB, and one can choose to pursue this line of reasoning when continuing to derive more beliefs. Nevertheless, it is also plausible to derive that $john$ is not a tax payer since there is an extension to prove this as well. The user ultimately decides which option to adopt further. We now turn our attention

to defaults applied to DLs.

2.5.2 Defaults Embedded in Description Logics

In the 90s, Baader and Hollunder attempted to embed Reiter's defaults into terminological formalisms (DLs) [11]. They consider the case where $\langle \mathcal{W}, \mathcal{D} \rangle$ represents a default theory in which $\mathcal{W}$ is a DL ($\mathcal{ALCF}$ in their investigation) KB and $\mathcal{D}$ is a finite set of open defaults.
In Reiter's treatment of open defaults, *Skolemization* is needed to make implicit individuals (introduced through existential quantification) explicit in the theory. To see why this is necessary, we give an example for the DL case:

Example 9 *Let* $\langle \mathcal{W}, \mathcal{D} \rangle = \langle \{\exists \mathsf{hasFriend}.\mathsf{Student}(john)\}, \{\frac{\mathsf{Student}:\neg\mathsf{EmployedPerson}}{\neg\mathsf{EmployedPerson}}, \frac{\mathsf{Student}:\exists\mathsf{hasFriend}.\mathsf{Student}}{\exists\mathsf{hasFriend}.\mathsf{Student}}\}\rangle$ *be a default theory.*

In Example 9 on Page 60, without an individual in our theory known to be a student, our defaults' prerequisite conditions are not met and the rules cannot "fire" to derive plausible inferences about students. However, the existential role quantifiers in $\mathcal{W}$ and $\mathcal{D}$ *do* introduce *implicit* student individuals into our theory.

The idea is to make these individuals explicit in the KB so that default reasoning can give the desired inferences. The way we accomplish this is through *skolemization*, i.e., the introduction of *skolem functions* to act as *placeholder* terms for the implicit individuals. See Reiter's' exposition [161, Section 7] for details of how the functions are defined.

Skolemizing $\mathcal{W}$, we obtain $\mathcal{W}' = \{\mathsf{hasFriend}(john, lily), \mathsf{Student}(lily)\}$ (by introducing the skolem constant $lily$). We also have to introduce a unary skolem function for the consequent of the default with existential quantifiers. We can derive that $lily$ is unemployed: $\neg\mathsf{EmployedPerson}(lily)$, and that $john$ has a friend who is unemployed: $(\exists\mathsf{hasFriend}.\ \neg\mathsf{EmployedPerson})(john)$.

We can also derive that $john$ has a friend, who has a friend that is a student $(\exists\mathsf{hasFriend}.\ \exists\mathsf{hasFriend}.\mathsf{Student})\ (john)$. □

Even though Skolemization helps to derive inferences that might otherwise have been lost, it is not without its problems. In particular, Skolemization can lead to strange and counter-intuitive inferences [11]:

Example 10 *Let* $\langle \mathcal{W}, \mathcal{D} \rangle = \langle \{(\exists \mathsf{hasSpouse}.\mathsf{Woman} \sqcup \mathsf{Bachelor})(john), \mathsf{hasSpouse}(john, sarah), \mathsf{Woman}(sarah)\}, \{\frac{:\neg \mathsf{Woman}}{\neg \mathsf{Woman}}\} \rangle$ *be a default theory.*

In order to identify the individuals that the default $\frac{:\neg \mathsf{Woman}}{\neg \mathsf{Woman}}$ can be applied to, we have to skolemize the ABox assertions. Skolemizing $\exists \mathsf{hasSpouse}.\mathsf{Woman}$ gives us two additional assertions: $\mathsf{hasSpouse}(john, david)$ and $\mathsf{Woman}(david)$.

Semantically, this narrows down our view to those interpretations of our ABox in which *john* has a spouse named *david* and *david* is a woman. But, because of the disjunction in $(\exists \mathsf{hasSpouse}.\mathsf{Woman} \sqcup \mathsf{Bachelor})(john)$, we find that it is perfectly acceptable for *david* not to be a woman in a model of our extended ABox.

Therefore, the "woman" default can fire and we can derive $\neg \mathsf{Woman}(david)$ and therefore, in order to satisfy $(\exists \mathsf{hasSpouse}.\mathsf{Woman} \sqcup \mathsf{Bachelor})(john)$ we have to accept that *john* is a bachelor - $\mathsf{Bachelor}(john)$, even though there is clear evidence in the ABox that he has a female spouse (*sarah* is a spouse of *john* and she is a woman). □

The problem in Example 10 on Page 61 is caused by not keeping track of what the skolem terms represent in the translated formulae. David Poole offers a solution to keep track of the skolem translation [154] using Hilbert's ϵ-symbol.
However, skolemization has other types of counter-intuitive inferences [11]:

Example 11 *Consider the two ABoxes:* $\mathcal{A}_1 = \{(\exists \mathsf{R}.(\mathsf{A} \sqcap \mathsf{B}))(a)\}$ *and* $\mathcal{A}_2 = \{(\exists \mathsf{R}.(\mathsf{A} \sqcap \mathsf{B}))(a), (\exists \mathsf{R}.\mathsf{A})(a)\}$ *and their skolemised versions,* $\mathcal{A'}_1 = \{\mathsf{R}(a, b), (\mathsf{A} \sqcap \mathsf{B})(b)\}$ *and* $\mathcal{A'}_2 = \{\mathsf{R}(a, c), (A \sqcap B)(c), \mathsf{R}(a, d), \mathsf{A}(d)\}$.

Notice that $\mathcal{A}_1$ is logically equivalent to $\mathcal{A}_2$. Given an open default $\{\frac{\mathsf{A}:\neg \mathsf{B}}{\neg \mathsf{B}}\}$, we can apply it to the individual d in $\mathcal{A'}_2$ to obtain $\neg \mathsf{B}(d)$ and, hence,

$(\exists \mathsf{R}.\neg \mathsf{B})(a)$. However, this is not a consequence of $\mathcal{A}'_1$ (even though $\mathcal{A}_1$ and $\mathcal{A}_2$ are logically equivalent). □

Lifschitz [122] adopts a model-theoretic approach to open defaults which avoids the problems caused by Skolemization. Unfortunately, his approach is not easily amenable to algorithmic construction. Moreover, there are "unexpected" inferences obtained in this approach for some special cases (see Section 3 of the work by Baader and Hollunder [11]).

On the other hand, considering the proposal of Baader and Hollunder, even without the problems of Skolemization, computing extensions for theories for a finite $\mathcal{W}$ and $\mathcal{D}$ still leads to undecidability [11, Section 4].

The authors therefore propose a restricted semantics in which defaults are only applied to explicitly mentioned individuals in the ABox. The obvious limitation is that one will not be able to derive some desired inferences, such as those discussed in Example 9 on Page 60. The good news is that with this restriction, decidability of computing extensions is retained and various algorithms have been presented to this end [11, Section 5].

One such algorithm is implemented by Kolovski, Parsia and Katz [108] in the OWL reasoner Pellet [182] and shown to have reasonable performance.

Despite the overall bleak outlook for defaults (in terms of being suitable for practical use), there are some who felt that the representation part of the formalism is very intuitive and still appropriate for integration into logics such as DLs. This persistence led to the recent work by Sengupta, Hitzler and Janowicz [177], who provided a new semantics for normal defaults.

Their resulting new notion of defaults (called *free defaults*) can be applied to implicit individuals while still retaining decidability, improving on the approach by Baader and Hollunder [11].

Another important issue surrounding defaults is the non-native treatment of *specificity*. Consider the following example:

Example 12 *Let* $\mathcal{W} = \{\mathsf{EmployedStudent} \sqsubseteq \mathsf{Student}, \mathsf{EmployedStudent}(john),$ $(\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top)(sarah)\}$ *and* $\mathcal{D} =$

$$\{\frac{\mathsf{Student}:\neg\mathsf{TaxPayer}}{\neg\mathsf{TaxPayer}}, \frac{\mathsf{EmployedStudent}:\mathsf{TaxPayer}}{\mathsf{TaxPayer}}, \frac{\mathsf{EmployedStudent}\sqcap\exists\mathsf{hasChild}.\top:\neg\mathsf{TaxPayer}}{\neg\mathsf{TaxPayer}}\}.$$

We obtain that *john* and *sarah* are plausibly tax payers, and also plausibly not tax payers. Since *sarah* and *john* are both general students and also employed students (with *sarah* even being a more specific type of employed student that has a child), it should be reasonable to *prefer* applying the more specific defaults to them.

That is, the defaults whose prerequisites most specifically characterise them. This would mean that the third default would be preferred to apply to *sarah* to derive that *sarah* is not a tax payer, and the second default would be preferred to apply to *john* to derive that he is a tax payer.

Extensions determined by applying the other defaults to these individuals should, arguably, be ignored. □

It is widely accepted that respecting specificity (of the kind exhibited in Example 12 on Page 62) should, in general, be a native attribute of defeasible reasoning formalisms. Since this property is not native to Reiter's default logic, there have been attempts to introduce appropriate versions of *priorities* on defaults that are taken into account when computing extensions [39, 37, 40, 12].

The approach by Baader and Hollunder [12], and those by Brewka [40], are the most compelling proposals to date for handling specificity and priorities in defaults. In general, priorities may be specified arbitrarily in these approaches, i.e., left up to the user to decide. However, the priority ordering on defaults induced by specificity of knowledge in $\mathcal{W}$, is advocated as an implicit presumption in reasoning.

Reiter's original formalism also natively endorses *transitivity*, which leads to undesired inferential behaviour [165]. If we have defaults expressing that university students are usually adults ($\frac{\mathsf{UniversityStudent}:\mathsf{Adult}}{\mathsf{Adult}}$), and that adults are usually employed ($\frac{\mathsf{Adult}:\mathsf{EmployedPerson}}{\mathsf{EmployedPerson}}$), then default reasoning admits the conclusion that university students are usually employed. That is, if we had an ABox assertion $\mathsf{UniversityStudent}(john)$ in our theory, then the theory can

be extended to include $\mathsf{EmployedPerson}(john)$.

Reiter proposes a solution to block transitivity by explicitly rewriting defaults to break the transitive links between them [165]. For our university student example, this approach would rewrite the default $\frac{\mathsf{Adult:EmployedPerson}}{\mathsf{EmployedPerson}}$ into $\frac{\mathsf{Adult:EmployedPerson} \sqcap \neg \mathsf{UniversityStudent}}{\mathsf{EmployedPerson}}$.

The obvious issue with this solution is that it causes the introduction of *non-normal* defaults, that is, defaults that do not share the nice computational properties that normal defaults do. For example, there is no guarantee that default theories with non-normal defaults will have an extension [161]. Fortunately, in many cases, Reiter's rewritings can avoid the introduction of non-normal defaults.

2.5.3 Discussion

While Reiter's default logic is very intuitive from a representational point-of-view, his formalism is not practically useful "out-of-the-box". Much of the formalism has had to be significantly revised in subsequent work to be suitable for DLs and to retain decidability. Some other relevant limitations include the following:

Computational Complexity and Performance: Decidability is not a guarantee of good practical performance. While many modifications to default logic are made to ensure decidability, there are very few attempts at proving tighter bounds on computational complexity. Moreover, implementation of default logic has a very sparse representation in the literature.

In fact, to the best of our knowledge, there is only one contemporary implementation by Kolovski, Parsia and Katz [108] with reasonable performance. Yet, since their work, there has been a definite wane in implementation and development of tools for representing Reiter-style defaults in DL-based ontologies. Hence, defaults have not been practically embraced, and thus, its expected practical performance in ontology development settings remains unclear.

Semantics: At the inception of default logic, Reiter himself admits that a significant drawback to the proposal is that it lacked a model theory. Lukaszewicz [126] later attempted to fill this gap and Etherington [66] built upon his foundation. Delgrande, Shaub and Jackson [60] use the foundation of Etherington and Lukaszewicz to define completely novel variants of default logic, modifying what they deem are unintuitive aspects of the formalism.

While these efforts are all noble additions to the default logic thread in defeasible reasoning, we argue that the kind of semantics they present are not instructive as to the meaning of default rules themselves. That is, a semantics in which we can interpret the meaning of a default rule in some interpretation or model structure, is not given. Indeed it is perhaps a difficult task to accomplish this seeing as defaults are encoded as inference rules.

We argue that this situation leads to the following unpleasant consequence of default reasoning research. That is, there is no general consensus on which variant (or combinations thereof) of Reiter's logic is the most suitable as a general nonmonotonic formalism. The picture is even more fuzzy in the context of DLs. Beyond the target of decidability (which has been met by various modifications to default logic), the semantic characterisation of the inferences that defaults give has not been standardised.

2.6 Minimal Knowledge and Negation as Failure (MKNF)

In the early 90s, Donini et al. [62] introduced *epistemic operators*, similar to those presented in Autoepistemic Logic [142], into terminological languages (the ancestors of DLs). This was in response to motivations by those such as Reiter [163], Lifschitz [123] and Levesque [119] that terminological languages should have epistemic querying capabilities. That is, viewing a KB as a set of statements about an external world, one should be able to pose queries about the external world that the KB is representing, as well as about what

the KB itself *knows* about the external world.

A simple example is given using $KB = \{(\exists \mathsf{owns.Dog})(susan), \mathsf{Dog} \sqsubseteq \mathsf{Pet}\}$ which represents that susan owns at least one dog, and that dogs are pets. One can ask the obvious query $KB \models (\exists \mathsf{owns.Pet})(susan)$? ("does susan own a pet?") and obtain the obvious answer "YES". However, one cannot ask queries of the form $KB \models (\exists \mathbf{K}\mathsf{owns.}\mathbf{K}\mathsf{Pet})(susan)$? ("is there something *known* to be owned by susan and *known* to be a pet?"), to which the system should respond "NO" because there is no such *evidence* in KB.

2.6.1 DLs of MKNF

Current DLs of MKNF [63, 107] enrich standard DLs with two epistemic operators on concepts, namely, **K** and **A**. The operator **K** (called the *minimal knowledge* operator) can be used in front of a concept C to obtain $\mathbf{K}C$, which intuitively represents all those objects which are *known* to be C's. In contrast, the operator **A** represents a *negation as failure* [54] modality which differs from **K**. The concept $\mathbf{A}C$ intuitively represents all those objects which can be *assumed* to be C's.

In order to capture default-like statements, $\mathbf{A}C$ is used with a negation in front of it as in $\neg\mathbf{A}C$, which represents the objects that *cannot* be assumed to be C's (notice the close correspondence with the meaning of a justification in a default presented in Section 2.5). In fact, the default $\frac{\mathsf{Student}:\neg\mathsf{TaxPayer}}{\neg\mathsf{TaxPayer}}$ can be encoded in the following DL axiom using the MKNF modalities discussed: $\mathbf{K}\mathsf{Student} \sqcap \neg\mathbf{A}\mathsf{TaxPayer} \sqsubseteq \neg\mathsf{TaxPayer}$.

Despite the similarities between MKNF and default logic, MKNF remains a more general formalism in which one can encode defaults [110] and possibly embed and study other nonmonotonic formalisms [125, 63]. In order to encode so-called "minimal knowledge" and the "default-like assumption" of negation as failure, the semantics of DLs with MKNF is built upon a more general interpretation structure $\langle \mathcal{I}, \mathcal{M}, \mathcal{N} \rangle$ where $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ is a standard DL interpretation and $\mathcal{M}$, $\mathcal{N}$ are *sets* of standard DL interpretations.

We state the concept language of $\mathcal{ALC}$ with MKNF ($\mathcal{ALCK}_{\mathcal{NF}}$) [63, Sec-

tion 2.2] here. Given a concept name A and role name R, one can define the concept language of $\mathcal{ALCK}_{\mathcal{NF}}$ as:

$$C ::= A \mid \top \mid \bot \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \exists R.C \mid \forall R.C \mid \mathbf{K}C \mid \mathbf{A}C$$

$\mathcal{ALCK}_{\mathcal{NF}}$ allows one to use the epistemic operators on roles as well which gives us the role language:

$$S ::= R \mid \mathbf{K}R \mid \mathbf{A}R$$

The semantics is analogous to that of standard DLs until we get to concepts (and roles) that include the epistemic operators. Atomic concepts and atomic roles are interpreted in $\mathcal{I}$ analogous to standard DLs. That is, for a given concept name A and role name R, $A^{\mathcal{I},\mathcal{M},\mathcal{N}} = A^{\mathcal{I}}$ and $R^{\mathcal{I},\mathcal{M},\mathcal{N}} = R^{\mathcal{I}}$. For the "non-MKNF" complex concepts we can extend this in the following way:

$$\begin{aligned}
&\top^{\mathcal{I},\mathcal{M},\mathcal{N}} = \Delta \\
&\bot^{\mathcal{I},\mathcal{M},\mathcal{N}} = \emptyset \\
&(\neg C)^{\mathcal{I},\mathcal{M},\mathcal{N}} = \Delta \backslash C^{\mathcal{I},\mathcal{M},\mathcal{N}} \\
&(C \sqcap D)^{\mathcal{I},\mathcal{M},\mathcal{N}} = C^{\mathcal{I},\mathcal{M},\mathcal{N}} \cap D^{\mathcal{I},\mathcal{M},\mathcal{N}} \\
&(C \sqcup D)^{\mathcal{I},\mathcal{M},\mathcal{N}} = C^{\mathcal{I},\mathcal{M},\mathcal{N}} \cup D^{\mathcal{I},\mathcal{M},\mathcal{N}} \\
&(\exists R.C)^{\mathcal{I},\mathcal{M},\mathcal{N}} = \{a \in \Delta \mid \text{there is a } b \in C^{\mathcal{I},\mathcal{M},\mathcal{N}} \text{ s.t. } (a,b) \in R^{\mathcal{I},\mathcal{M},\mathcal{N}}\} \\
&(\forall R.C)^{\mathcal{I},\mathcal{M},\mathcal{N}} = \{a \in \Delta \mid \text{for all } b, (a,b) \in R^{\mathcal{I},\mathcal{M},\mathcal{N}} \implies b \in C^{\mathcal{I},\mathcal{M},\mathcal{N}}\}
\end{aligned}$$

It is apparent that the above definition does not capture novel semantics and is analogous to standard DLs. However, when we interpret concepts (and roles) containing epistemic operators, we notice the first references to the sets $\mathcal{M}$ and $\mathcal{N}$. Intuitively, $\mathcal{M}$ and $\mathcal{N}$ are introduced to interpret the minimal knowledge and negation as failure notions on concepts (hence the assigned letters to name these sets):

$$(\mathbf{K}C)^{\mathcal{I},\mathcal{M},\mathcal{N}} = \bigcap_{\mathcal{J}\in\mathcal{M}} C^{\mathcal{J},\mathcal{M},\mathcal{N}}$$
$$(\mathbf{A}C)^{\mathcal{I},\mathcal{M},\mathcal{N}} = \bigcap_{\mathcal{J}\in\mathcal{N}} C^{\mathcal{J},\mathcal{M},\mathcal{N}}$$

That is, $(\mathbf{K}C)^{\mathcal{I},\mathcal{M},\mathcal{N}}$ denotes the intersection of the C's in each interpretation of the given "minimal knowledge" set of interpretations $\mathcal{M}$ (the key is that the interpretation domain Δ is fixed across interpretations). $(\mathbf{A}C)^{\mathcal{I},\mathcal{M},\mathcal{N}}$ is interpreted analogously, but on the "negation as failure" set of interpretations $\mathcal{N}$. The meaning of "minimal knowledge" and "negation as failure", as well as the difference between **K** and **A**, in this context, becomes more clear when we define what constitutes a *model* for an $\mathcal{ALCK}_{\mathcal{NF}}$ KB.

The axiom language of $\mathcal{ALCK}_{\mathcal{NF}}$ is the same as that of $\mathcal{ALC}$. We have subsumptions in the TBox and concept and role assertions in the ABox. Satisfaction in an interpretation $\langle\mathcal{I},\mathcal{M},\mathcal{N}\rangle$ is analogous to standard DLs. We say that an axiom α is satisfied in a structure $\langle\mathcal{M},\mathcal{N}\rangle$ if α is satisfied in $\langle\mathcal{I},\mathcal{M},\mathcal{N}\rangle$ for each $\mathcal{I}\in\mathcal{M}$. We extend this satisfaction definition to TBoxes, ABoxes and, more generally, to $\mathcal{ALCK}_{\mathcal{NF}}$ KBs in an analogous way to standard DLs. We now define a model for an $\mathcal{ALCK}_{\mathcal{NF}}$ KB:

Definition 4 (Model for $\mathcal{ALCK}_{\mathcal{NF}}$ KB) *Let $\mathcal{K}$ be an $\mathcal{ALCK}_{\mathcal{NF}}$ KB. A set of interpretations $\mathcal{M}$ is a model for $\mathcal{K}$ if the structure $\langle\mathcal{M},\mathcal{M}\rangle$ satisfies $\mathcal{K}$ and, for each set of interpretations $\mathcal{M}'\supset\mathcal{M}$,$\langle\mathcal{M}',\mathcal{M}\rangle$ does* not *satisfy $\mathcal{K}$.*

Therefore, Definition 4 on Page 68 enforces a notion of *maximality* to the set of interpretations that satisfy the KB. Coupled with the interpretation of **K** and **A** defined above, we can see that $\mathbf{K}C$ intuitively captures the C's that we know to be present in each (and every) interpretation that satisfies our knowledge. One can see that this maximisation of the set of interpretations that satisfy our knowledge actually captures a notion of *minimal* knowledge about C's.

Recall that one can encode defaults in DLs of MKNF. For example, one can represent the information that "by default, students do not pay taxes" using the axiom $\mathbf{K}\mathsf{Student}\sqcap\neg\mathbf{A}\mathsf{TaxPayer}\sqsubseteq\neg\mathsf{TaxPayer}$. This MKNF encoding

intuitively means "if something is *known* to be a student and it cannot be *assumed* to be a tax payer, then it is *not* a tax payer".

The semantics of DLs with MKNF state that "*known* to be" means appearing in all applicable interpretations, and that "cannot be *assumed* to be" means there is at least one applicable interpretation in which this assumption is false. An employed student introduced in the KB who *does* pay taxes will then be interpreted as conflicting with the $\neg\mathbf{A}\mathsf{TaxPayer}$ predicate, and thus be allowed to retain its tax paying property.

When translating prerequisite-free defaults into MKNF axioms, a new concept has to be introduced into the translation [63, Section 3.1] to align with the semantics of Baader et al. [11] (recall that, in the latter work of Baader et al., defaults are only applied to individuals explicitly mentioned in the ABox to avoid counter-intuitive inferences).

In addition to default representation, DLs with MKNF are useful for representing *integrity constraints* [163]. Integrity constraints are statements about what the KB *should* know (what it is required to know). For example, one can add the integrity constraint $\mathbf{K}\mathsf{Employee} \sqsubseteq \mathbf{A}\mathsf{Male} \sqcup \mathbf{A}\mathsf{Female}$ [63, Example 3.2] to a KB indicating that "any known employees should be known to be either male or female". This constraint will then force inconsistency of the KB if an employee is added without explicitly mentioning their gender.

In summary, linking up with our philosophy on imperfect knowledge (Section 2.3), we recall that incompleteness is unfortunately a general property of knowledge. Since DLs with MKNF try to model this incompleteness more abstractly than other approaches to defeasible reasoning, by distinguishing between general constraints and explicitly *known* facts, they are, representationally speaking, very powerful formalisms for reasoning with exceptions.

2.6.2 Discussion

DLs with MKNF are indeed very expressive formalisms and add a new dimension to knowledge representation and reasoning with DLs. However, as we now discuss, it seems that "with great power comes great complexity".

Computational Complexity: In order to obtain decidable reasoning for $\mathcal{ALCK}_{\mathcal{NF}}$, Donini et al. [63, Section 4] adopted the strategy of reducing reasoning in $\mathcal{ALCK}_{\mathcal{NF}}$ to reasoning in $\mathcal{ALC}$. They tried to show that one can represent the models for an $\mathcal{ALCK}_{\mathcal{NF}}$ KB by a finite set of $\mathcal{ALC}$ KBs, in which case reasoning can be executed on the latter and thus inherit decidability.

However, they found that such a representation does not exist in general. Instead they identified a restricted subset of $\mathcal{ALCK}_{\mathcal{NF}}$ KBs for which such representations exist. While this subset might capture a large subset of realistic KBs, it definitely omits other perfectly reasonable ones (see Definitions 4.2 and 4.11 in their exposition [63] which give the criteria for representable KBs). Even accepting these restrictions, though, a first upper bound for complexity is identified as 3EXPSPACE [63, Section 4.3] which is discouraging when compared with other defeasible reasoning approaches.

Representational Complexity: We also argue that, from the perspective of an ontology engineer, the usage of epistemic features in DLs has the potential to be cognitively complex. It can be argued that the operators themselves are intuitively simple to understand and, considering there are just two extra operators, it is not a major departure from the feature set of standard DLs.

However, in some cases it may lead to cognitive burden in modelling. Specifically, it is sometimes unclear which operator, **K** or **A** should be used. For example, it is unclear whether the integrity constraint $\mathbf{K}\mathsf{Employee} \sqsubseteq \mathbf{A}\mathsf{Male} \sqcup \mathbf{A}\mathsf{Female}$ is equivalent to $\mathbf{K}\mathsf{Employee} \sqsubseteq \mathbf{K}\mathsf{Male} \sqcup \mathbf{K}\mathsf{Female}$.

2.7 Defeasible Logic

Defeasible logic is a formalism consolidated by Donald Nute [149] sharing similar core ideas to that of John Pollock concerning defeasible reasoning [153]. The key representational elements of the logic are default-like sentences called

defeasible rules (one can also represent counterpart *strict rules*), as well as specialised rules called *defeaters* which specify conditions under which defeasible rules may be overridden.

In order to better understand the intuition behind these constructs, it is helpful to consider the philosophical underpinnings of defeasible logic. An important perspective that both Pollock and Nute take in their treatises is that whenever we represent knowledge as logical sentences, we have implicit justifications for "believing" these statements that we write down. They explain that some of these reasons are "conclusive" whereas others aren't. This allows one to draw a distinction between strict rules whose justifications are all conclusive, and defeasible rules that have some inconclusive ones.

2.7.1 Basic Defeasible Logic

Defeasible logic is generally built upon propositional logic but can, in principle, be built upon richer languages. A defeasible logic theory or KB consists of five components: a set of *facts*, a set of defeasible rules, a set of strict rules, a set of defeaters and a *priority relation* (an acyclic and transitive partial order in most formalisations) among defeasible rules.

Facts are statements about the domain represented by a literal (e.g. "john is a student" represented by $\mathsf{Student}(john)$). A defeasible (resp. strict) rule r has the form $r : A(r) \implies B(r)$ (resp. $r : A(r) \rightarrow B(r)$) consisting of its unique label r. $A(r)$ is called the *antecedent* or *body* of r and represents a set of literals and $B(r)$ is a single literal called the *consequent* or *head* of r. Rules may have no body but they always have a head.

An example of a defeasible rule is "Employed students generally pay taxes" written as $\mathsf{EmployedStudent}(x) \implies \mathsf{TaxPayer}(x)$. An example of a strict rule is "All students have a student ID" written as $\mathsf{Student}(x) \rightarrow \mathsf{hasID}(x)$. Defeaters, as mentioned earlier, are caveats that can prevent the application of rules. Hence, it is important to note that defeaters do not aid in *drawing* conclusions, rather, they are used to *prevent* some. Similar to rules, they have the form $A(r) \rightsquigarrow B(r)$ where $A(r)$ is called the *antecedent*

or *body* of r and represents a set of literals and $B(r)$ is a single literal called the *consequent* or *head* of r.

Defeaters are required to have both a head and a body. An example of a defeater is "Employed students who are actually university teaching assistants might not be obliged to pay taxes" written as $\mathsf{EmployedStudent}(x)$, $\mathsf{TeachingAssistant}(x) \rightsquigarrow \neg\mathsf{TaxPayer}(x)$. We also may need to specify priorities among rules. For example, given rules $r1 : \mathsf{Student}(x) \implies \neg\mathsf{TaxPayer}(x)$, $r2 : \mathsf{EmployedStudent}(x) \rightarrow \mathsf{Student}(x)$ and $r3 : \mathsf{EmployedStudent}(x) \implies \mathsf{TaxPayer}(x)$, and a fact $f1 : \mathsf{EmployedStudent}(john)$, there is a conflict between $r1$ and $r3$ when applied to $f1$. If we apply $r1$ then we derive $\neg\mathsf{TaxPayer}(john)$, while applying $r3$ gives us the contradictory conclusion $\mathsf{TaxPayer}(john)$. It makes sense to prefer applying $r3$ over $r1$ since $r3$ is applied to the more specific information about $john$ (that he is employed). We can specify this preference using $r3 > r1$.

Just like default logic, defeasible logic follows a fixed-point construction in deriving inferences. Recall that the inference mechanism of default logic worked by applying default rules in an arbitrary order to derive extensions. During the application of rules in a particular ordering, other rules may become blocked and a fixed point is reached when no other rules can be applied. The goal is to compute all such fixed points or *extensions* of the given KB, or sometimes to just determine if a given formula appears in any of these extensions.

In defeasible logics, the inference mechanism is, in principle, slightly different because of the added defeater constructors. Additionally, given a propositional atom q and a defeasible logic KB $\mathcal{K}$, we have four different types of conclusions one can draw about q: (1) "$+\Delta q$" which means that q is *definitely* provable from $\mathcal{K}$ (i.e., purely from the facts and strict rules) (2) "$-\Delta q$" which means that q can be shown to *not* be definitely provable from $\mathcal{K}$ (3) "$+\delta q$" which means that q is *defeasibly* provable from $\mathcal{K}$ and (4) "$-\delta q$" which means that q can be shown to *not* be defeasibly provable from $\mathcal{K}$.

Conclusions are thus meta-theoretical statements about provability and

not part of the language of defeasible logic. Suppose we are given a defeasible logic theory $\mathcal{K} = \langle F, R, D, > \rangle$ where F is the set of facts, R is the set of rules (both defeasible and strict), D is the set of defeaters and $>$ is the priority relation on D. Then the first two (strict) conclusion types are derived through the following inference rules:

- $+\Delta q$: q can be definitely proved from $\mathcal{K}$ if either $q \in F$ or there is a strict rule $r \in R$ with q as the head and, for all antecedent atoms a in r, a can be definitely proved from $\mathcal{K}$.
- $-\Delta q$: q can be shown to *not* be definitely provable from $\mathcal{K}$ if $q \notin F$ *and* for each strict rule $r \in R$ with q as the head, there is an antecedent atom a in r s.t. a can be shown to *not* be definitely provable from $\mathcal{K}$.

The two rules have recursive definitions with a straightforward meaning (the terminating case of recursion is when we find compatible knowledge in our *fact* base). The first rule says that we can definitely prove q if it is already contained in our fact base, or it appears as the consequent of a strict rule whose antecedents are all either satisfied or, when recursing on strict rules who have these antecedents as consequents, we eventually reach a termination point with all antecedents of a strict rule present in our fact base.

The second rule is just the complement of the first rule: we can show that q is not definitely provable from $\mathcal{K}$ if it is not satisfied (not in our fact base) *and* for each strict rule in which q is the consequent there is at least one antecedent a in this rule that is not satisfied (not in our fact base) or, if all antecedents a are satisfied for some rules, then we recurse on these by taking those rules with a as consequent, examining their antecedents to eventually terminate when we find an antecedent that is *not* in the fact base.

The two *defeasible* inference rules are more complex considering that we have to take into account defeaters (we have to consider "opposing" chains of reasoning) and priority relations among rules. As we shall see, these rules also "interact" or "depend" on each other.

- $+\delta q$: q is defeasibly provable from $\mathcal{K}$ if either $+\Delta q$ w.r.t. $\mathcal{K}$, or (1) there is a rule $r \in R$ (defeasible or strict) with q as the consequent and for each antecedent a of r, $+\delta a$ w.r.t. $\mathcal{K}$, and (2) $-\Delta(\neg q)$ w.r.t. $\mathcal{K}$, and for each $s \in R \cup D$ s.t. $\neg q$ is the consequent of s, either ($3a$) there is an antecedent a of s such that $-\delta a$ w.r.t. $\mathcal{K}$ or ($3b$) there is a rule $t \in R$ (defeasible or strict) with q as the consequent s.t. for each antecedent a of t, $+\delta a$ w.r.t. $\mathcal{K}$ and t has priority over s ($t > s$).

- $-\delta q$: q can be shown to *not* be defeasibly provable from $\mathcal{K}$ if $-\Delta q$ w.r.t. $\mathcal{K}$ *and* (1) for each rule $r \in R$ (defeasible or strict) with q as consequent, there is an antecedent a of r s.t. $-\delta a$ w.r.t. $\mathcal{K}$, or (2) $+\Delta(\neg q)$ w.r.t. $\mathcal{K}$, or there is an $s \in R \cup D$ s.t. $\neg q$ is the consequent of s such that ($3a$) for each antecedent a of s, $+\delta a$ w.r.t. $\mathcal{K}$ and ($3b$) for each $t \in R$ (defeasible or strict) with q as the consequent, either there is an antecedent a of t s.t. $-\delta a$ w.r.t. $\mathcal{K}$ or t does *not* have priority over s ($t \not> s$).

We run through the inference rule $+\delta q$ and skip discussion on $-\delta q$ (the latter is just the negation of the former). Firstly, if q is *definitely* provable (derivable purely from the strict rules and facts) then it is intuitive that the additional defeasible rules and defeaters should not interfere with this. That is, it is intuitive to infer that it q is (*defeasibly*) provable even considering the additional defeasible constructors as well.

Now supposing that we are given a defeasible logic KB $\mathcal{K}$ from which we *cannot* derive q purely from the strict information. This means we have to argue based on the defeasible information. Let us suppose that there are no "attacks" on concluding q using the defeasible rules. Then, Point (1) of the definition for $+\delta q$ says that, considering the defeasible rules with q as a consequent, if we can find one whose antecedents are all *defeasibly* provable (at this point in the inductive definition, *defeasibly* provable here reduces to the termination case of *definitely* provable) then we can conclude that q is also *defeasibly* provable.

Now one has to consider possible "attacks" on concluding q. Point (2) has to eliminate the attack which forces us to *definitely* prove $\neg q$ (from the *strict* information only) because this would clearly contradict with defeasibly proving q. In addition, we have to consider all the possible *defeasible* rules and defeaters that could possibly lead us to prove $\neg q$.

This is done in Point (3) and in order to endorse q we need to ensure that, for each defeasible rule with $\neg q$ in the head (allowing us to conclude $\neg q$), *either*: (a) there is an antecedent of such a rule that is *not* defeasibly provable (here we refer to the complementary rule $-\delta q$ whose terminating case is $-\Delta q$), or (b) there is another defeasible rule whose head is q (allowing us to conclude q) and it has higher priority then the one leading us to the contradictory case $\neg q$.

We have given a very brief presentation of the mechanics of defeasible logic including its language constructs and inference mechanism. More recently, a very interesting investigation in defeasible logic by Antoniou et al. [3] revealed that much of the constructs in defeasible logic are "syntactic sugar". That is, these constructs do not add any expressive power to the logic.

The authors found that for any *well-formed* defeasible logic KB (the priority relation is acyclic and only defined on rules with conflicting or complementary heads), the KB can be reduced purely to a set of strict and defeasible rules (absorbing the facts, defeaters and priority relation). Therefore, representationally speaking, this logic closely resembles default logic when restricted to normal defaults.

The main advantage of formalisms resembling that of defeasible logic are its low computational complexity. It is been shown to have linear complexity in the size of the defeasible logic theory [131]. However, the price that we pay for such efficiency is that defeasible logic embodies a relatively "shallow" reasoning paradigm, especially when compared to logics such as DLs. Therefore, there will be some desirable intuitive inferences that defeasible logic will be too weak to capture. Since our formalisms of interest are DLs, we have to discuss the state-of-the-art in combining defeasible logic with DLs.

2.7.2 Combining Defeasible and Description Logics

Defeasible logic was originally designed as a proof-theoretic rule system with a propositional base language used for antecedents and consequents in rules. In terms of semantics, such characterisations (which came in a variety of flavours [82, 130]) came much later in the development.

Thus defeasible logic, in principle, may be superimposed over a formalism such as Description Logic. There have been various approaches that accomplish this [1, 81, 195].

Wang et al. [195] and Antoniu et al. [1] took similar approaches by proposing to place defeasible logic rules "on top of" a DL ontology. In the rules one is allowed to express *dl-literals* (literals representing references to concepts in the underlying ontology) *only* in the antecedent of rules. Essentially, the inference mechanism follows the same structure as presented in Section 2.7.1 but when we have to check provability of antecedents that are dl-literals (representing DL concepts), this is done using a DL reasoner with respect to the underlying ontology. Thus, an applicable DL reasoner needs to be embedded in the defeasible logic reasoner.

The approach by Governatori [81] appears a more natural combination of defeasible logic with DLs. The set of ABox statements in the DL KB constitutes the facts in a defeasible logic theory, the subsumptions are translated into strict rules and defeasibility is introduced by adding standard defeasible rules to the theory. A KB in this hybrid theory has the structure $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, R, > \rangle$ where $\mathcal{A}$ is an ABox, $\mathcal{T}$ is a TBox, R is a set of defeasible rules and $>$ is the priority relation among the rules in R.

Governatori chooses $\mathcal{ALC}^{-}$ to be the logic for $\mathcal{A}$ and $\mathcal{T}$. This restricted version of $\mathcal{ALC}$ omits existential role restrictions from consideration. Therefore, since the only additional constructs over propositional logic that need to be considered are *universal* role restrictions, he formulates additional inference rules (four types mirroring those presented in Section 2.7.1) for deriving knowledge of the form $(\forall R.C)(a)$ where R is a role name, C is a concept and a is an individual. In particular, only explicitly mentioned individuals

in $\mathcal{A}$ are considered in the inference mechanisms. The *defeasible* rule types are defined as follows:

- $+\delta(\forall R.C)(a)$: "all a's R-successors are C's" is *defeasibly* provable from $\mathcal{K}$ if for each individual b mentioned in $\mathcal{A}$, either $-\delta R(a,b)$ (see specification of $-\delta q$ in Section 2.7.1), or $+\delta C(b)$ (see specification of $+\delta q$ in Section 2.7.1).

- $-\delta(\forall R.C)(a)$: "all a's R-successors are C's" can be shown to *not* be defeasibly provable from $\mathcal{K}$ if there is an individual b mentioned in $\mathcal{A}$ s.t. $+\delta R(a,b)$ *and* $-\delta C(b)$.

Notice that the rules above *loosely* simulate the semantics of universal role restrictions in DLs. Pothipruk and Governatori [155] later extend this approach to allow expression of existential restrictions in the KB but they do not formulate inference rules for defeasible derivation of such information. In summary, the combination of defeasible logic and DLs follows the basic model of placing rules on top of an underlying DL ontology and using the inference mechanisms of defeasible logic to draw conclusions using these rules.

The structure of rules and the inference mechanisms have to be modified to take into account the additional richness of DLs and some restrictions need to be placed on the expressive power of the combined system in order to prevent counterintuitive inclusions.

Representationally speaking, defeasible logic appears to have a close resemblance with default logic. However, Antoniou et al. [2] show that a fairly representative version of defeasible logic can be embedded in default logic, suggesting that default logic may be a more general and inferentially powerful formalism. Still, standard defeasible logic remains the most computationally efficient nonmonotonic formalism in recent years, having a linear worst case complexity in the size of the defeasible theory.

In addition to the presented integration efforts with DLs, defeasible logic has also been applied in various other settings. Most notably in *logic programming* [4] for which, it can be argued, it is more naturally suited. Even though

defeasible logic was not designed with a formal semantics in mind, various semantics have been proposed subsequently. An argumentation-theoretic semantics has been proposed by Governatori et al. [82] and a model-theoretic semantics has been presented by Maher [130].

2.7.3 Discussion

Even though the combinations of defeasible logic and DLs attempt to closely model the semantics of DLs in inference rules (e.g. for universal role restrictions), the resulting mechanism falls short in a number of ways from the inferential power of DLs. This is because the inference rules alone cannot make use of the richness of the DL and therefore the combined approach inherits the same inferential weaknesses of standard defeasible logic.

Here is a simple instance of this weakness: given the set of DL subsumptions $\{C \sqsubseteq D, \neg C \sqsubseteq D\}$ (where C and D are DL concepts) one would be able to conclude, using DL reasoning, that $\top \sqsubseteq D$ (everything in the domain is a D). However, translating this set into the corresponding strict rules in $\{C \rightarrow D, \neg C \rightarrow D\}$ (where C and D are literals) we find that, using defeasible logic, one cannot derive D ($+\Delta D$).

We have mentioned an extension of this approach [155] which allows expression of existential restrictions in rules. However, there is no accompanying mechanism for deriving relevant inferences from these. In DLs, for example, given the TBox $\{\mathsf{Student} \sqsubseteq \neg\mathsf{Employed}\}$ and the following ABox $\{(\exists\mathsf{hasFriend}.\mathsf{Student})(john)\}$, one would be able to conclude that "john has a friend who is not employed" $((\exists\mathsf{hasFriend}.\neg\mathsf{Employed})(john))$. However, in the translation to defeasible logic we have the rule $r : \mathsf{Student}(x) \rightarrow \neg\mathsf{Employed}(x)$ and the ABox $\{(\exists\mathsf{hasFriend}.\mathsf{Student})(john)\}$ from which we cannot derive the inference that $+\Delta(\exists\mathsf{hasFriend}.\neg\mathsf{Employed})(john)$ because there is no applicable inference rule.

In summary, defeasible logic was, by design, intended to be very computationally efficient. As a result, its overt proof-theoretic flavour placed much emphasis on justifying inference through proof strategy and argument.

In the end, the investigation of enriching its language of antecedents and consequents, and taking such structure into account during inference, remains under-developed. Finally, this has led to a more "shallow" reasoning paradigm relative to other approaches.

2.8 Preferential Reasoning

In the late 80s to early 90s, Kraus, Lehmann and Magidor (KLM) [111, 117, 116] took to studying nonmonotonic reasoning from the standpoint of the consequence relations that it *should* induce. This was quite a different perspective from which to proceed when compared to many other approaches to defeasible reasoning.

The basic motivation for this is that nonmonotonic formalisms define approaches for deriving *plausible* inferences when knowledge is considered *imperfect* (see Section 2.3), and since there may be various notions of plausibility that one can define, there may be multiple definitions for entailment. KLM argued that, even though there can be multiple notions of plausibility, the consequence relations that each notion induces should at least satisfy some basic logical properties in order to be called "rational" entailment.

In other words, they defined a standard for "rational" nonmonotonic entailment [117, Section 3], so that future nonmonotonic formalisms can be evaluated against this standard to understand their logical merit.

The resulting *preferential approach*, as it is often called, is this general framework that we choose as the basis for our entire investigation in this thesis. Throughout this thesis, unless otherwise stated, when we refer to the term preferential, we are referring to the general framework introduced by KLM to investigate defeasible reasoning. In the normal cases we refer to this general framework by the phrases: "preferential approach", "preferential framework" or "preferential context" etc.

We motivate fully in Section 2.10 why we choose the preferential approach, over the other candidates presented in this chapter, towards the goals

of this thesis. For now, we start by giving a basic overview of the preferential approach in the context of propositional logic. Thereafter, in Section 2.8.2, we discuss the state-of-the-art in efforts to generalise this approach to DLs. Section 2.8.2 is particularly important because it serves as the theoretical basis of the work presented in this thesis.

2.8.1 Propositional Foundations

Nonmonotonic reasoning, as the name suggests, does not, in general, satisfy the logical property of monotonicity. While there may also be various other logical properties that nonmonotonic reasoning does not satisfy, KLM diverted the focus by studying nonmonotonic reasoning from the perspective of the consequence relations it induces [111], in order to identify logical properties that it *should* actually satisfy.

Their initial characterisations of nonmonotonic consequence relations considered KBs represented in classical propositional logic [111]. That is, given a KB of sentences represented in classical propositional logic, the idea was to characterise "suitable" nonmonotonic entailment relations to be able to conclude what classical propositional sentences would follow from this KB. The suitability of the entailment relations was determined by investigating the logical properties that they should satisfy. This can be seen as a meta-level approach for introducing nonmonotonic reasoning into propositional logic.

However, KLM later took a different approach to introducing nonmonotonic reasoning in propositional logic. In this later work [117], the idea was to enrich propositional logic with a kind of "defeasible implication" (i.e., a connective on the object level), and then given a KB represented in the new logic, to agree upon logical properties that this *connective* should satisfy, in order to induce suitable nonmonotonic entailment relations for the new logic. This perspective can be seen as an object-level approach for introducing nonmonotonic reasoning into propositional logic and is the one that we explore further in this thesis.

In this latter investigation, KLM introduce a new connective $\mid\!\sim$ into

propositional logic called *conditional implication* (also called *defeasible implication*). The intended, intuitive meaning of the sentence $p \mathrel{|\!\sim} q$ is that p *usually* implies q, where p and q are classical propositional formulas. They use *conditional KBs* to encode knowledge about the domain of interest.

Definition 5 (Conditional KB) *A* conditional KB *is a pair* $\langle \mathcal{T}_{PL}, \mathcal{D}_{PL} \rangle$ *where* $\mathcal{T}_{PL}$ *is a finite set of classical propositional formulas and* $\mathcal{D}_{PL}$ *is a finite set of conditional implications.*

Within this context, they agreed on a series of logical postulates (called the KLM postulates) that they argue the connective $\mathrel{|\!\sim}$ should satisfy if it is to induce suitable nonmonotonic consequence relations [117, Section 3]. We restate the properties below for completeness:

$$\text{(Ref) } \alpha \mathrel{|\!\sim} \alpha \qquad \text{(LLE) } \frac{\models \alpha \leftrightarrow \beta,\ \alpha \mathrel{|\!\sim} \gamma}{\beta \mathrel{|\!\sim} \gamma}$$

$$\text{(And) } \frac{\alpha \mathrel{|\!\sim} \beta,\ \alpha \mathrel{|\!\sim} \gamma}{\alpha \mathrel{|\!\sim} \beta \wedge \gamma} \qquad \text{(Or) } \frac{\alpha \mathrel{|\!\sim} \gamma,\ \beta \mathrel{|\!\sim} \gamma}{\alpha \vee \beta \mathrel{|\!\sim} \gamma}$$

$$\text{(RW) } \frac{\alpha \mathrel{|\!\sim} \beta,\ \models \beta \rightarrow \gamma}{\alpha \mathrel{|\!\sim} \gamma} \qquad \text{(CM) } \frac{\alpha \mathrel{|\!\sim} \beta,\ \alpha \mathrel{|\!\sim} \gamma}{\alpha \wedge \beta \mathrel{|\!\sim} \gamma}$$

$$\text{(RM) } \frac{\alpha \mathrel{|\!\sim} \gamma,\ \alpha \not\mathrel{|\!\sim} \neg\beta}{\alpha \wedge \beta \mathrel{|\!\sim} \gamma}$$

The property (Ref) stands for Reflexivity and it obviously captures that α should be plausibly derivable from itself. (LLE) stands for Left Logical Equivalence and captures that if α and β are indistinguishable, and γ can be plausibly derived from α, then γ should be plausibly derivable from β as well. The (And) postulate endorses that $\beta \wedge \gamma$ is plausibly derivable from α if both β and γ are plausibly derivable from α separately. The (Or) postulate says that γ is plausibly derivable from $\alpha \vee \beta$ if it is plausibly derivable from α and β separately.

(RW) stands for Right Weakening and captures that if β is plausibly derivable from α and γ is *classically* derivable from β, then γ is plausibly derivable

from α (notice that Right Weakening captures a kind of transitivity property for $\mid\!\sim$). The properties of Cautious Monotonicity (CM) and Rational Monotonicity (RM) are perhaps the most interesting of the postulates because they concern weakened versions of the property of *Classical* Monotonicity. Obviously, we would not like $\mid\!\sim$ to satisfy classical monotonicity (because we are interested in defining nonmonotonic consequence relations), but if it did, it would satisfy the following property:

$$\text{(M)}\ \frac{\alpha \mid\!\sim \gamma}{\alpha \wedge \beta \mid\!\sim \gamma}$$

(M) says that if γ is plausibly derivable from α, then γ is plausibly derivable from $\alpha \wedge \beta$ for any formula β (whatever additional information I have, it cannot force me to retract my conclusions). Monotonicity enforces *cumulative* knowledge. That is, one can only build upon existing knowledge and cannot retract old knowledge. Cautious Monotonicity weakens classical monotonicity by stating that if I can plausibly derive γ from α, I can plausibly derive γ from $\alpha \wedge \beta$ provided that I can also plausibly derive γ from β separately.

Rational Monotonicity is a related property which weakens classical monotonicity by endorsing $\alpha \wedge \beta \mid\!\sim \gamma$ if γ is plausibly derivable from α and one *cannot* plausibly derive $\neg\beta$ from α. Both (CM) and (RM), then, define some extra *safety conditions* under which we can endorse cumulative knowledge.

At this point, while we have described the postulates that the *connective* $\mid\!\sim$ should satisfy, we have not discussed how it is possible to evaluate a particular *consequence relation* against these postulates to determine if it satisfies them or not.

It is clear that given a KB $\mathsf{K} = \{\alpha_1 \mid\!\sim \beta_1, \ldots, \alpha_n \mid\!\sim \beta_n\}$, there are in general many closures of K. That is, there are many supersets K' of K that satisfy the KLM postulates. For example given a particular K' I can evaluate whether it satisfies the (And) rule by verifying for each $\alpha_1 \mid\!\sim \beta_1 \in \mathsf{K}'$ and $\alpha_1 \mid\!\sim \beta_2 \in \mathsf{K}'$ that $\alpha_1 \mid\!\sim \beta_1 \wedge \beta_2 \in \mathsf{K}'$. We motivate why any nonmonotonic DL should satisfy the KLM postulates in Chapter 3.

With an aim towards defining "rational" entailment regimes for nonmonotonic formalisms, KLM use the term *rational consequence relation* to refer to a consequence relation that satisfies all the KLM postulates. They also identified the most "conservative" rational consequence relation, that is, the one endorsing the fewest (positive) plausible inferences. They coined this relation the *Rational Closure* [117, Section 5] and provided a model-theoretic description of Rational Closure using a "preference" style of semantics based on the proposal by Yoav Shoham [180].

KLM also define entailment regimes based on consequence relations that satisfy all the KLM postulates except for RM, such as *Preferential entailment* [117, Definition 2.8] and *Ranked entailment* [117, Section 4][6]. However, considering the pragmatic goals of this thesis, these proposals are not suitable to consider because they define consequence relations which are monotonic [117, Sections 2.4 and 4.2], and therefore defeats the purpose of a defeasible reasoning paradigm which has to be able to revise knowledge.

Therefore, for this thesis, we focus on Rational Closure as a starting point for developing practical systems for defeasible reasoning in the context of DLs. The semantic foundation of preferential reasoning, and hence Rational Closure, rests on the notion of a *ranked interpretation* [117, Definition 3.8] (and we present a generalisation of this definition, for DLs, in the next section). KLM also precisely define what an *exception* [117, Definition 2.20] is, in terms of ranked interpretations, and we present a generalisation of this definition in the next section for DLs. The notion of exception will prove to be central to, and of vital importance for, generalising the KLM algorithm for computing Rational Closure to the DL setting. It also proves to be essential for giving novel (but related) algorithms for computing defeasible entailment in the preferential framework for DLs.

[6]It was demonstrated by KLM that the notions of preferential entailment and ranked entailment are actually identical [117, Section 4.2]

2.8.2 Description Logic Foundations

In this section we present the state-of-the-art when it comes to the theoretical foundation of preferential reasoning in DLs (or, equivalently, *preferential DLs*). All relevant definitions and characterisations are adapted from the work of Giordano et al. [71] and Britz et al. [41].

In preferential DLs, a new kind of subsumption relation $\sqsubset\!\!\!\sim$ is employed in the language. Using $\sqsubset\!\!\!\sim$, one can formulate subsumption statements of the form $C \sqsubset\!\!\!\sim D$ called *defeasible subsumption statements* or just *defeasible subsumptions*, where C and D are classical $\mathcal{ALC}$ concepts. $C \sqsubset\!\!\!\sim D$ is read as "C is *usually* subsumed by D". For our purposes, since we are interested in $\mathcal{ALC}$, our base language consists of the standard concept and axiom language for $\mathcal{ALC}$ (Section 2.1.1) together with defeasible subsumption. Now, in order to interpret defeasible subsumption statements we have to enrich the semantics for classical DLs.

In classical DLs [10], the semantics is built upon first order interpretations. These interpretations vary on the elements which appear in the interpretation domain ($\Delta^{\mathcal{I}}$) and the manner in which we assign terms to these elements (and *pairs* of these elements in the case of roles) - defined by an interpretation function ($\cdot^{\mathcal{I}}$). In the preferential context, an additional component on which the interpretations can vary, is introduced. This component represents the manner in which we can order the elements of the domain, using a *modular ordering*[7] ($\prec_{\mathcal{I}}$).

Definition 6 (Modular Order) *Given a set X, a relation $\prec \subseteq X \times X$ is* modular *if there is an ordering function $o : X \longrightarrow \mathbb{N}$ s.t. for every $x, y \in X$, $x \prec y$ if and only if $o(x) < o(y)$.*

[7]Historically, this ordering started out as a partial order as a theoretical exercise and led to the definition of *preferential* interpretations. Later, KLM defined the notion of *ranked interpretation* which considered totally ordered, *well-founded* sets (those having a minimal element) to which to map the elements of the domain, defining a "ranking" or total order on the elements. In this thesis, we do not take such a general stance, we pick a *specific* well-founded set - the natural numbers - to define our ranked interpretations.

Intuitively, the ordering on domain elements can be seen to reflect the "typicality" or "normality" of the element w.r.t. all possible properties that it can possess. We elaborate more on this later in this section. Interpretations whose ordering on domain elements respects Definition 6 on Page 84, are known as *ranked* interpretations [117].

Definition 7 (Ranked Interpretation) *A ranked interpretation is a structure* $\mathcal{R} := \langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}}, \prec^{\mathcal{R}} \rangle$*, where* $\langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}} \rangle$ *is a DL interpretation (which we denote by* $\mathcal{I}_{\mathcal{R}}$ *and refer to as the classical interpretation associated with* $\mathcal{R}$*), and* $\prec^{\mathcal{R}}$ *is a modular ordering on* $\Delta^{\mathcal{R}}$*.*[8]

Modular orderings allow us to "rank" the elements in the domain by assigning a natural number to each element representing their "level" in the model. However, function o in Definition 6 on Page 84 still allows for "levels" in our ranked interpretation that do not contain any elements.

For example, let $S = \{x_1, x_2, x_3, x_4\}$ be a set of elements, and let $\prec$ be a modular ordering on the elements of S defined s.t. $x_1 \prec x_2$, $x_2 \prec x_4$ and $x_1 \prec x_3$. Then we can define an ordering function o (as described in Definition 6 on Page 84) s.t. $o(x_1) = 0$, $o(x_2) = 2$, $o(x_3) = 2$ and $o(x_4) = 4$ which respects $\prec$. Notice that we have no elements in S on level 1 and level 3 of our ordering. In order to remedy this situation we define the *rank* of a domain element in a ranked interpretation using a *ranking function* which eliminates these empty levels:

Definition 8 (Rank of a Domain Element) *Let* $\mathcal{R} = \langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}}, \prec_{\mathcal{R}} \rangle$ *be a ranked interpretation, the* rank *of an element* $x \in \Delta^{\mathcal{R}}$*, denoted by* $rk_{\mathcal{R}}(x)$*, is the length of the chain* $x_0 \prec_{\mathcal{R}} \ldots \prec_{\mathcal{R}} x$ *where* x_0 *is any* minimal *element in the ordering* $\prec_{\mathcal{R}}$*.*

We can also refer to the top-most (highest) rank in a ranked interpretation:

[8] Given $X \subseteq \Delta^{\mathcal{R}}$, we denote the set $\{x \in X \mid \text{for every } y \in X, y \not\prec_{\mathcal{R}} x\}$ with $\min_{\prec_{\mathcal{R}}}(X)$. Intuitively, $\min_{\prec_{\mathcal{R}}}(X)$ denotes the most "typical" elements of X. That is, the elements in X that are the most minimal in the partial ordering.

Definition 9 (Maximal Rank) *Given a ranked interpretation $\mathcal{R}$, the* maximal rank *(denoted by $max(\mathcal{R})$) of $\mathcal{R}$ is $rk_{\mathcal{R}}(x)$ for some $x \in \Delta^{\mathcal{R}}$ s.t. there is no $y \neq x$ s.t. $y \in \Delta^{\mathcal{R}}$ and $rk_{\mathcal{R}}(y) > rk_{\mathcal{R}}(x)$. Similarly, given a non-empty set of ranked interpretations $\mathscr{R} = \{\mathcal{R}_1, ..., \mathcal{R}_n\}$ where $n > 0$, the* maximal rank *for $\mathscr{R}$, denoted by $max(\mathscr{R})$, is the largest number in the set $\{max(\mathcal{R}_1), ..., max(\mathcal{R}_n)\}$.*

We point out that the bottom-most or lowest rank of any ranked interpretation is always 0 unless otherwise stated. We sometimes refer to this rank as the *minimal rank* of a ranked interpretation. Given the definition of the rank of an element in a ranked interpretation, we can also define the *rank of a concept* in a ranked interpretation as follows.

Definition 10 (Rank of a Concept in a Ranked Interpretation) *Given a ranked interpretation $\mathcal{R}$, and a concept C, let $\mathcal{R}_C = \{rk_{\mathcal{R}}(x) \mid x \in C^{\mathcal{R}}\}$. Then, the* rank *of C w.r.t. $\mathcal{R}$ (denoted by $rk_{\mathcal{R}}(C)$) is the smallest number in $\mathcal{R}_C$. If $C^{\mathcal{R}} = \emptyset$ then we say that C has* no rank *w.r.t. $\mathcal{R}$ or equivalently that C has* infinite rank *w.r.t. $\mathcal{R}$ (denoted by $rk_{\mathcal{R}}(C) = \infty$).*

Given the semantic foundation we have laid, we can interpret defeasible subsumption statements of the form $C \sqsubseteq\!\!\!\!\sim\, D$ (see Figure 2.2 for a graphical representation) using ranked interpretations. Technically, $C \sqsubseteq\!\!\!\!\sim\, D$ is satisfied in a ranked interpretation $\mathcal{R}$ if $\min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \subseteq D^{\mathcal{R}}$ (see Definition 7 on Page 85) and we denote this by $\mathcal{R} \Vdash C \sqsubseteq\!\!\!\!\sim\, D$. We also say that $\mathcal{R}$ is a *ranked model* for $C \sqsubseteq\!\!\!\!\sim\, D$ (analogous to the classical DL case).

We can also interpret classical DL subsumptions $C \sqsubseteq D$ in ranked interpretations in the usual way. That is, a ranked interpretation $\mathcal{R} = \langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}}, \prec_{\mathcal{R}} \rangle$ satisfies a subsumption $C \sqsubseteq D$ if $\langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}} \rangle$ satisfies it, and we denote this by $\mathcal{R} \Vdash C \sqsubseteq D$ ($\mathcal{R}$ is a *ranked model* for $C \sqsubseteq D$). One can construct *defeasible KBs* consisting of both classical and defeasible DL subsumptions.

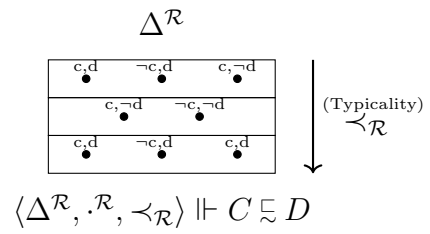


Figure 2.2: Satisfaction of a defeasible subsumption by a ranked interpretation.

Definition 11 (Defeasible KB)
A defeasible KB *is a structure* $\langle \mathcal{T}, \mathcal{D} \rangle$ *where* $\mathcal{T}$ *is a (possibly empty) finite set of classical DL subsumptions, and* $\mathcal{D}$ *is a (possibly empty) finite set of* $\mathcal{ALC}$ *defeasible subsumptions (called a* defeasible TBox *or* DTBox*).*

We can extend satisfaction in a ranked interpretation to defeasible KBs as well. A defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ is satisfied by a ranked interpretation $\mathcal{R}$ if $\mathcal{R}$ satisfies each subsumption in $\mathcal{T} \cup \mathcal{D}$. In such cases we say that $\mathcal{R}$ is a *ranked model* for $\langle \mathcal{T}, \mathcal{D} \rangle$. Based on Definition 10 on Page 86 we can extend the notion of the rank of a concept to the context of a defeasible KB.

Definition 12 (Rank of a Concept w.r.t. a Defeasible KB)
Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ *and a concept* C*, let* $\mathscr{R}$ *be the set of all ranked models for* $\langle \mathcal{T}, \mathcal{D} \rangle$ *and let* $\mathscr{R}_C = \{rk_{\mathcal{R}}(C) \mid \mathcal{R} \in \mathscr{R}\}$*. Then, the rank of* C *w.r.t.* $\langle \mathcal{T}, \mathcal{D} \rangle$ *(denoted by* $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C)$*) is the smallest number in* $\mathscr{R}_C$*.*

As discussed earlier, there is no unique version of entailment for nonmonotonic formalisms in general. Some formalisms such as Circumscription even give the user of the reasoning system a degree of control in defining entailment by supplying a circumscription pattern. In the preferential context we also have multiple possible entailment proposals. Indeed, it can be argued that different proposals can be suitable for different applications. Before we explore this issue, we define the notion of entailment in the preferential context that is analogous to classical DL entailment.

Recall that classical DLs consider *all* models for a given classical KB in order to define entailment. The analogous proposal in the preferential context is to consider what follows in all *ranked* models for the given defeasible KB. This version of entailment is called *ranked entailment* and is a generalisation of ranked entailment in the propositional setting [117, Section 4].

Definition 13 (Ranked Entailment) *Given a defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$ *and a defeasible subsumption* $C \sqsubseteq\!\!\!\!\sim D$, $C \sqsubseteq\!\!\!\!\sim D$ *is* ranked entailed *by* $\langle \mathcal{T}, \mathcal{D} \rangle$, *written as* $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \sqsubseteq\!\!\!\!\sim D$, *if each ranked model for* $\langle \mathcal{T}, \mathcal{D} \rangle$ *is also a ranked model for* $C \sqsubseteq\!\!\!\!\sim D$.

It turns out that ranked entailment is monotonic in the DL case as well [41, Section 4], and this results in an entailment regime which is unsuitable for defeasible reasoning which is supposed to be able to retract old knowledge. However, before we explore the problem of defining appropriate versions of entailment for preferential DLs, we have to formalise the central principle of *exception* in this context, since this thesis is concerned with DL reasoning in the presence of exceptions.

We recall that, in contrast to a standard DL subsumption $C \sqsubseteq D$, which we read as "all C's are D's", the reading of the defeasible subsumption $C \sqsubseteq\!\!\!\!\sim D$, is that "the most *typical* C's are D's" (see Definition 7 on Page 85 and accompanying footnote). It is the ordering on elements in a ranked interpretation that allows us to isolate *typical* elements. The semantic paradigm which this approach captures is very intuitive because it is one which we as humans often employ (albeit in an implicit way). Consider the following example:

Example 13 *Suppose that* Bob *and* John *are mechanics. If we don't have any other information then as humans we may implicitly regard* Bob *and* John *as* typical *mechanics and assign to them properties that a typical mechanic may possess. For example we may conclude that* Bob *and* John *both work in a workshop. However, we may later discover that, while* Bob *works from a workshop,* John *is actually a* mobile *mechanic and only repairs machinery at the clients premises - which means he does not work from a workshop. One*

may say that Bob *is more typical than* John *w.r.t. the property of possessing a workshop. What this means is that* John *is more* exceptional *than* Bob *w.r.t. the same property. But what if we consider a different property of a typical mechanic? We may consider a typical mechanic to have one or more types of machinery that they* specialise *in. If we find that* John *indeed has a specialisation in motorboats but that* Bob *does not have a specialisation in any specific equipment types then we implicitly consider* John *to be more typical than* Bob *in this context.* □

Example 13 on Page 88 raises an often asked question about typicality in the context of ranked interpretations: given a ranked interpretation, what is the motivation for its chosen typicality ordering of domain elements? It is an interesting question because when it comes to the components of a standard first order interpretation, it is not often motivated why one would choose a particular interpretation domain rather than another.

Similarly, it is also seldom motivated why one would assign (or not assign) a specific term from our vocabulary to an element of the domain. However, when it comes to typicality orderings, a motivation for a particular ordering choice is often demanded.

The preferential DLs that we consider give the following explanation: given two elements x and y in some interpretation domain of a ranked interpretation, if we choose to interpret x as less exceptional (more typical) than y (written $x \prec y$), we mean that we assume that there is a function or black-box that is able to consider every conceivable property of x and y, aggregate the exceptionality of x and y w.r.t. these properties, and in the end assign a natural number to x and y representing their "degree of exceptionality".

Thus $x \prec y$ (x is less exceptional than y) if the degree of exceptionality of x is less than that of y. Essentially, the typicality orderings can vary across ranked interpretations because evaluating typicality w.r.t. all conceivable properties could be a subjective task.

Returning to Example 13 on Page 88, we show how the framework of ranked interpretations handles multiple conflicting typicality orderings. If

we *only* had the constraint in our KB that typical mechanics work in a workshop ($\mathsf{Mechanic} \sqsubseteq \exists \mathsf{hasWorkshop}.\top$) then, assuming that Bob is a *prototypical* mechanic i.e., $\mathsf{Bob} \in \min_{\prec_{\mathcal{R}}}(\mathsf{Mechanic}^{\mathcal{R}})$ for some ranked model $\mathcal{R}$ for our knowledge, then John *has* to be considered more exceptional (higher in the ordering) than Bob in $\mathcal{R}$.

Let us suppose that this is *not* the case. That is, perhaps Bob is considered more exceptional than John, or they are considered equally exceptional. If they are equally exceptional then we have a conflict with our knowledge, because this tells us that John is also a typical mechanic and should possess a workshop (whereas Example 13 on Page 88 tells us otherwise). If our ordering dictates that Bob is *more* exceptional than John, then we have a conflict with our assumption that Bob is a prototypical mechanic because it means that John is lower down in our ordering.

Essentially, this means that the ranked interpretations whose ordering functions dictate that John and Bob are equally exceptional, or that Bob is more exceptional than John, are incompatible with our knowledge.

Similarly, if we *only* had the constraint that typical mechanics have a specialisation ($\mathsf{Mechanic} \sqsubseteq \exists \mathsf{hasSpecialisation}.\top$) then Bob is more exceptional than John (assuming John is a prototypical mechanic). But what if we have to satisfy *both* constraints? Suppose our background knowledge is that typical mechanics work in workshops *and* that typical mechanics have at least one specialisation. Consider three possible ranked models for this knowledge:

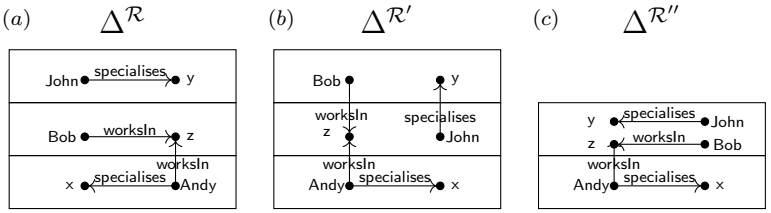


Figure 2.3: Multiple valid typicality orderings of ranked interpretations.

It is clear that if our background knowledge about mechanics is correct, then there must exist at least one typical mechanic who satisfies both our con-

straints. If there isn't, then we would be in a logically incoherent situation and would have to revise or retract our statements (if no typical mechanics exist then no mechanics exist). Since Example 13 on Page 88 makes mention only of Bob and John, and both these individuals are missing one of the required properties, we have to conclude that there must be a third individual. Let's call him Andy and he is a very typical mechanic i.e. he possesses both required properties by working in a workshop and specialising in automobiles.

Both Bob and John can then be seen as exceptional w.r.t. the prototypical mechanic Andy. But how do we decide who is more exceptional between Bob and John? The answer is that we don't have to because Andy satisfies our knowledge; Bob and John are exceptional to Andy so the exceptionality distinction between them does not matter. The typicality orderings in the ranked models (a), (b) and (c) in Figure 2.3 are all valid for our knowledge and so we have to consider all of them.

A strong advantage of preferential logics is the behaviour represented in Figures 2.2 and 2.3 where the ranked interpretations satisfy that the most typical C's (lowest in the ordering) are also D's, but still allows some C's that are not as typical (higher up in the ordering) to not be D's. This is the ability to gracefully cope with *exceptions* - which is something that standard DLs cannot.

Many fields such as biology and medicine deal with information which holds in general but is fallible under exceptional circumstances. Given this setting, biologists and medical professionals still have to draw conclusions and make decisions based upon this information. Preferential DLs are developed for applications of this kind. This leads us to a formal definition for *concept exceptionality* in preferential DLs, which is a straightforward generalisation of the definition given by KLM in the propositional case [117, Definition 2.20].

Definition 14 (Concept Exceptionality) *Given a defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$ *and a concept* C*,* C *is* exceptional *w.r.t.* $\langle \mathcal{T}, \mathcal{D} \rangle$ *if* $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq \neg C$*. Each subsumption* $C \sqsubseteq D \in \mathcal{D}$ *(for any concept* D*) is also said to be* exceptional *w.r.t.* $\langle \mathcal{T}, \mathcal{D} \rangle$*.*

That is, a concept C is exceptional w.r.t. a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ if there is no ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. there is an element x on the most typical (bottom-most) level of the model that also belongs to $C^{\mathcal{R}}$ (the most typical elements in our models are not C's). This intuitively captures that C's cannot be considered "prototypical" or "stereotypical" in any sense in our models. Consider the following example.

Example 14 *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a defeasible KB where:*
$\mathcal{T} = \{\mathsf{MobileMechanic} \sqsubseteq \mathsf{Mechanic}, \mathsf{GeneralMechanic} \sqsubseteq \mathsf{Mechanic}\}$ *and*
$\mathcal{D} = \{\mathsf{Mechanic} \mathrel{\sqsubset\!\!\!\sim} \exists \mathsf{hasWorkshop}.\top, \mathsf{Mechanic} \mathrel{\sqsubset\!\!\!\sim} \exists \mathsf{hasSpecialisation}.\top,$
$\mathsf{MobileMechanic} \mathrel{\sqsubset\!\!\!\sim} \neg\exists \mathsf{hasWorkshop}.\top, \mathsf{GeneralMechanic} \mathrel{\sqsubset\!\!\!\sim} \neg\exists \mathsf{hasSpecialisation}.\top\}$

It is easy to see that one cannot "realise" a MobileMechanic or GeneralMechanic object on the bottom level of a ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$. In other words, the concepts MobileMechanic and GeneralMechanic are exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$. This behaviour is intuitive because these concepts can be seen as exceptional types of Mechanic that do not possess all the properties that one would typically associate with the latter. □

We now define *concept unsatisfiability* in the context of preferential DLs. The definition is analogous to the one for classical DLs. Interestingly, concept unsatisfiability in this context also characterises a special case of concept exceptionality.

Definition 15 (Ranked Concept Unsatisfiability) *Given an $\mathcal{ALC}$ defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and an $\mathcal{ALC}$ concept C, C is* ranked unsatisfiable *w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ if $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \mathrel{\sqsubset\!\!\!\sim} \bot$. We refer to C as* totally exceptional *w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$.*

Notice that if $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \mathrel{\sqsubset\!\!\!\sim} \bot$ then $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \mathrel{\sqsubset\!\!\!\sim} \neg C$. That is, total exceptionality is logically stronger than exceptionality. Also, total exceptionality (or equivalently ranked unsatisfiability) is analogous to classical unsatisfiability. This is easy to see because $C \mathrel{\sqsubset\!\!\!\sim} \bot$ says that typical C's do not exist, which corresponds to saying that C's do not exist (it is straightforward to see that for any ranked interpretation $\mathcal{R}$, $\mathcal{R} \Vdash C \mathrel{\sqsubset\!\!\!\sim} \bot$ if and only if $\mathcal{R} \Vdash C \sqsubseteq \bot$).

On the other hand, $\top \not\sqsubseteq \neg C$ is logically weaker. It allows C's to exist but prescribes that the most typical *things* in our domain are not C's. We would like to draw a distinction between exceptionality and its special case of total exceptionality. The need to distinguish between these will become more apparent when we present algorithms for computing defeasible entailment in Chapter 4. Hence, separating out the totally exceptional cases we can refer to concepts that are *normally exceptional*.

Definition 16 (Concept Normal Exceptionality)
Given an $\mathcal{ALC}$ defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and an $\mathcal{ALC}$ concept C, C is normally exceptional *w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ if C is exceptional but* not *totally exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$.*

We have presented an overview of the semantic foundation for preferential reasoning in DLs. Linking up with the pragmatic goals of this thesis, which is to build practical reasoning systems upon this foundation, we have to give an overview on the state-of-the-art in this regard.

2.8.3 Rational Closure for Description Logics

Recall that ranked entailment (Definition 13 on Page 88), in the DL setting, is monotonic and therefore unsuitable for defeasible reasoning. However, in the propositional setting, KLM presented the landmark entailment regime for preferential reasoning, Rational Closure, which is not monotonic and enjoys some elegant mathematical properties [117, Section 5].
Furthermore, KLM argue that Rational Closure is the most conservative notion of nonmonotonic entailment that can be considered rational [117, Thesis 1.1]. That is, a rational nonmonotonic formalism may endorse more inferences than Rational Closure but it should at least endorse all Rational Closure's inferences.

Because of this, the main efforts to generalise preferential reasoning to DLs have converged on generalising the definition of Rational Closure to DLs. Indeed, this is probably the best starting point for defining defeasible

entailment in preferential DLs. KLM give several definitions for Rational Closure in their seminal work, in terms of rational consequence relations [117, Definition 5.7], ranked model semantics [117, Section 5.7] and the rank of a (propositional) formula w.r.t. a conditional KB [117, Theorem 5.17].

The main efforts to characterise Rational Closure for DLs opted to generalise the definition based on the *rank of a formula*. Giordano et al. [71, Definition 21] and Britz et al. [41, Definition 14] generalise the rank of a propositional formula to the rank of a concept in DLs but *w.r.t. a defeasible KB* (recall that Definition 10 on Page 86 is a definition *w.r.t. a ranked interpretation*).

Since KLM formulated their definition *operationally* (because the semantic perspective was not their initial angle of investigation), the definitions of Giordano et al. [71, Definition 21] and Britz et al. [41, Definition 14] mirror this convention. Before we restate this definition we have to define the notions of *exceptional subset* [71, Definition 19] and *exceptional subset sequence* [71, Definition 20] of a defeasible KB.

Definition 17 (Exceptional Subset) *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{D} \rangle$ be a defeasible KB. Then, the* exceptional subset *of $\mathcal{K}$ (denoted by $\mathcal{E}(\mathcal{K})$) is defined as $\langle \mathcal{T}, \mathcal{D}_{exc} \rangle$ where $\mathcal{D}_{exc} = \{C \sqsubset\!\!\sim D \in \mathcal{D} \mid C \text{ is exceptional w.r.t. } \langle \mathcal{T}, \mathcal{D} \rangle\}$.*

Definition 17 on Page 94 allows us to define a *sequence* of exceptional subsets of a defeasible KB.

Definition 18 (Exceptional Subset Sequence)
Given a defeasible KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{D} \rangle$, the exceptional subset sequence *of $\mathcal{K}$ is the sequence E_0, E_1, ..., E_n ($n \geq 0$) where $E_0 = \mathcal{K}$ and $E_i = \mathcal{E}(E_{i-1})$ for $1 \leq i \leq n$.*

Informally, we start with the original KB E_0 and obtain the set of all defeasible subsumptions in E_0 that are exceptional. This subset will constitute E_1 and we recurse on E_1 to obtain E_2 and so on. The process is executed until we reach a *fixed point* in which one of two things will happen for some

i: (1) $E_i = E_{i-1}$ (all the subsumptions in E_i are exceptional) or (2) $E_i = \emptyset$ (none of the subsumptions in E_i are exceptional). Given this exceptionality sequence we can give an operational definition for the rank of a concept (corresponding with the semantic one given in Definition 12 on Page 87) as follows [71, Definition 21].

Definition 19 (Rank of a Concept w.r.t. a Defeasible KB) *Given a concept C, a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and its exceptional subset sequence $E = \{E_0, \ldots, E_n\}$. Then, the rank of C w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ (denoted by $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_p}(C)$) is the smallest $0 \leq i \leq n$ s.t. C is* not *exceptional w.r.t. E_i. If C is exceptional w.r.t. E_i for each $1 \leq i \leq n$ then C has infinite rank denoted by $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_p}(C) = \infty$.*

The correspondence between the semantic definition (Definition 12 on Page 87) and procedural definition (Definition 19 on Page 95) for the rank of a concept has been shown by Giordano et al. [71, Proposition 13]. Therefore, the following definition for Rational Closure (agreed upon by current generalisations of preferential reasoning for DLs), is actually a semantic definition (see the work by Giordano et al. [71, Definition 22] and of Britz et al. [41, Definition 15]).

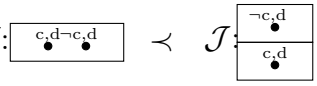
Definition 20 (Rational Closure of a Defeasible KB) *Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and a defeasible subsumption $C \sqsubseteq D$, $C \sqsubseteq D$ is in the* Rational Closure *of $\langle \mathcal{T}, \mathcal{D} \rangle$ (denoted by $\langle \mathcal{T}, \mathcal{D} \rangle \models_{rational} C \sqsubseteq D$) if $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C) < rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C \sqcap \neg D)$ or $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C) = \infty$.*

While Definition 20 on Page 95 arguably captures an intuitive and reasonable form of entailment in the preferential context, it is still recommended to study its characterisation in terms of ranked models in detail, in order to determine its logical merit.
Recall that ranked entailment (i.e., all ranked models for our KB) was too strong a notion of entailment. Rational Closure, then, narrows its view to a subset of these models. It is important to note that this subset is not

some arbitrary one, and indeed, upon studying Rational Closure from other perspectives (such as its consequence relation and its definition in terms of the rank of a concept) one can appreciate that its model-theoretic characterisation would, at the very least, be interesting if not suitable as a general entailment regime. The current DL versions of Rational Closure point to a characterisation in terms of *minimal* ranked models [71, Definition 25] similar to the propositional case [117, Section 5.7].

Minimal ranked models are defined by placing a partial ordering on the ranked models for the KB - this is in *addition* to the partial ordering on the elements of the domain (see Figure 2.4 for an example). The minimal

$$\langle \mathcal{T}, \mathcal{D} \rangle = \langle \emptyset, \{C \sqsubseteq D\} \rangle$$



$\mathcal{I}$ is a minimal ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$

Figure 2.4: Ordering ranked models in pursuit of the minimal ones.

ranked models in the partial order are those in which there is no element of the domain that can be moved to a more typical level in the strata (i.e. if it can be moved, then it is not possible without violating at least one axiom in the KB). We adapt the definition for *minimal ranked models* here from the work of Giordano et al. [71].

Definition 21 (Minimal Ranked Model) *Let* $\mathcal{R} = \langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}}, \prec_{\mathcal{R}} \rangle$ *and* $\mathcal{R}'$ $= \langle \Delta^{\mathcal{R}'}, \cdot^{\mathcal{R}'}, \prec_{\mathcal{R}'} \rangle$ *be two ranked interpretations. We say that* $\mathcal{R}$ *is more* preferred *than* $\mathcal{R}'$ *if:*

- $\Delta^{\mathcal{R}} = \Delta^{\mathcal{R}'}$,
- $\cdot^{\mathcal{R}} = \cdot^{\mathcal{R}'}$,
- *for each* $x \in \Delta^{\mathcal{R}}$, $rk_{\mathcal{R}}(x) \leq rk_{\mathcal{R}'}(x)$ *and there is a* $y \in \Delta^{\mathcal{R}}$ *s.t.* $rk_{\mathcal{R}}(y) < rk_{\mathcal{R}'}(y)$.

Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ *and a ranked model* $\mathcal{R}$ *for* $\langle \mathcal{T}, \mathcal{D} \rangle$*, we say that* $\mathcal{R}$ *is a* minimal *ranked model for* $\langle \mathcal{T}, \mathcal{D} \rangle$ *if there is no other ranked model* $\mathcal{R}'$ *for* $\langle \mathcal{T}, \mathcal{D} \rangle$ *s.t.* $\mathcal{R}'$ *is more preferred than* $\mathcal{R}$*.*

Further to Definition 21 on Page 96, Giordano et al. [71, Theorem 7] show that there is a minimal ranked model $\mathcal{R}$ that is *canonical* [71, Definition 24 and Definition 25] for each consistent $\mathcal{ALC}$ defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, that defines Rational Closure. That is, $\mathcal{R} \Vdash C \mathrel{\sqsubset\!\!\sim} D$ if and only if $\langle \mathcal{T}, \mathcal{D} \rangle \models_{rational} C \mathrel{\sqsubset\!\!\sim} D$.

In fact, Giordano et al. show that there always exists such a canonical ranked model for any consistent $\mathcal{ALC}$ defeasible KB and there will always be *more than one* of these. Nevertheless, they showed that any *one* of these models is sufficient to characterise Rational Closure. We are going to repeat the definition of minimal canonical ranked model here, since it is essential for characterising Rational Closure in $\mathcal{ALC}$. However, before we do this, we introduce some new terms.

Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and a query axiom $\delta = C \mathrel{\sqsubset\!\!\sim} D$, we refer to the set of all concepts and subconcepts occurring in $\langle \mathcal{T}, \mathcal{D} \rangle$ and δ, together with their negations, as the *concept universe* for $\langle \mathcal{T}, \mathcal{D} \rangle$ and δ (always denoted by $\mathscr{C}_{\langle \mathcal{T}, \mathcal{D} \rangle, \delta}$). Given the notion of concept universe, we can define the subsets of $\mathscr{C}_{\langle \mathcal{T}, \mathcal{D} \rangle, \delta}$ that are *compatible* with $\langle \mathcal{T}, \mathcal{D} \rangle$.

Definition 22 (Compatible Concepts for a Defeasible KB) *Let* $\langle \mathcal{T}, \mathcal{D} \rangle$ *be a defeasible KB and let* $\delta = C \mathrel{\sqsubset\!\!\sim} D$ *be a query axiom. Then,* $\{C_1, \ldots, C_n\} \subseteq \mathscr{C}_{\langle \mathcal{T}, \mathcal{D} \rangle, \delta}$ *is* compatible *with* $\langle \mathcal{T}, \mathcal{D} \rangle$ *if* $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r C_1 \sqcap C_2 \sqcap \ldots \sqcap C_n \sqsubseteq \bot$*.*

Definition 22 on Page 97 leads us to a succinct definition for a minimal canonical ranked model for some defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$.

Definition 23 (Minimal Canonical Ranked Model) *Let* $\langle \mathcal{T}, \mathcal{D} \rangle$ *be a defeasible KB and* $\mathcal{R}$ *be a minimal ranked model for* $\langle \mathcal{T}, \mathcal{D} \rangle$*. Then,* $\mathcal{R}$ *is a minimal* canonical *ranked model for* $\langle \mathcal{T}, \mathcal{D} \rangle$ *if for each compatible subset* $\{C_1, \ldots, C_n\} \subseteq \mathscr{C}_{\langle \mathcal{T}, \mathcal{D} \rangle, \delta}$*, there is an* $x \in (C_1 \sqcap C_2 \sqcap \ldots \sqcap C_n)^{\mathcal{R}}$*.*

Giordano et al. [71, Theorem 7] have shown that the Rational Closure of each $\mathcal{ALC}$ defeasible KB, that has a ranked model, can be characterised by a single canonical ranked model. The semantics for Rational Closure is then defined by this minimal canonical ranked model.

From a pragmatic perspective, the first attempt at a procedure for computing a defeasible entailment regime similar to Rational Closure (in the DL case) was the effort of Casini and Straccia [51] for $\mathcal{ALC}$. A substantial advantage of this syntactic procedure was that it was composed entirely of classical DL entailment checks. In contrast, Giordano et al. [75] present a tableau calculus for constructing ranked models in their preferential extension of $\mathcal{ALC}$, however, both these constructions do not compute exactly the notion of Rational Closure we define in Definition 20 on Page 95.

Notwithstanding, all existing procedures for computing Rational Closure in the literature, that are based on classical DL decision steps, are variants of the syntactic procedure by Casini and Straccia. In fact, an updated variant of this procedure is presented in Chapter 4 together with some alternative forms of defeasible entailment for preferential DLs.

2.8.4 Rational Extensions of an ABox

In the literature for preferential DLs, the general arc has been to first "lift" the theoretical foundation of KLM, from the propositional setting, to DLs. Thereafter, the next step is to characterise and develop algorithms for Rational Closure for the *TBox* (i.e., TBoxes that contain defeasible subsumptions) and finally, to address entailment for ABoxes. Because of the difference in structure of DLs (over classical propositional logic), ABox approaches lead to a unique set of challenges.

In these approaches the task is to consider an extended notion of defeasible KB (we extend the structure $\langle \mathcal{T}, \mathcal{D} \rangle$ to $\langle \mathcal{T}, \mathcal{D}, \mathcal{A} \rangle$ i.e., including a classical ABox $\mathcal{A}$) and to define what defeasible entailment means in this context. That is, for the two main types of assertions $C(a)$ and $R(a, b)$ in $\mathcal{ALC}$, we have to define defeasible counterparts for these assertions so that we

can ask if *a presumably* can fall under the concept C (in the former assertion) and if a is *presumably* related to b via R (in the latter), from $\langle \mathcal{T}, \mathcal{D}, \mathcal{A} \rangle$.

Giordano et al. [71, Section 3.3] and Casini et al. [50] presented both semantic definitions and algorithmic constructions, based on these definitions, for ABox Rational Closure in $\mathcal{ALC}$. The basic intuition behind these efforts is to consider the individuals mentioned in the ABox as typical as possible (considering those ranked models in which the objects referred to by the mentioned individual names are as low down as possible in the ordering). A positive attribute of the semantics is that it has a similar flavour to the semantics for TBoxes mentioned in the previous section and thus inherits some attractive properties of the latter.

It can also be shown that the worst-case computational complexity of their ABox procedure is the same as for TBox entailment [50, Proposition 3]. Although, upon an analysis of the algorithms in relation to one another, and having experience of the nature and size of ABoxes in real-world ontologies, it is expected that ABox entailment would not perform as well, in practice, as the TBox procedures it is based on.

Despite the presence of a foundation for reasoning with ABoxes, there are still some gaps to be filled in the area. The issue of how to handle *implicit* individuals in the ABox, for example, arises in this context as well (echoing the situation in default reasoning - see Example 9 on Page 60). We refer the reader to the provided references for more detailed information about preferential reasoning for ABoxes. We remind the reader that the primary concern of this thesis is a practical foundation for preferential reasoning with *TBoxes*. Therefore, we do not explore ABox entailment any further in this thesis.

2.8.5 Discussion

Because of its relative immaturity to other forms of defeasible reasoning in DLs, the preferential approach (and specifically its suitability for defeasible reasoning in DLs) requires more investigation. As we have discussed in this

section, the preferential approach is a *framework* for defeasible reasoning. The framework can be used for a variety of purposes: to study the logical merit of nonmonotonic formalisms, to compare nonmonotonic formalisms from the standpoint of their consequence relations and, of course, to build entailment proposals on top of the framework (there is not just one notion of entailment in the preferential framework).

Rational Closure defines a very useful defeasible entailment regime for DLs and is, in our view, just as strong a candidate for TBox defeasible reasoning as any other defeasible formalism in the literature (if not superior in some respects). However, for some situations where more adventurous reasoning is required, its inferential behaviour can be viewed as slightly weak. Nevertheless, even in such settings, it is a suitable starting point from which to investigate more adventurous entailment regimes for preferential DLs.

Alternatives to Rational Closure: In the efforts to lift the preferential framework to DLs, Rational Closure was the first notion of entailment to be lifted to the DL case. This is indeed very reasonable since Rational Closure is the most conservative notion enjoying all the desirable KLM properties. However, in some settings a more adventurous reasoning behaviour is required and Rational Closure can sometimes be too skeptical to give back some of these more adventurous inferences. We give more detail about this issue in Chapter 4.

Application to Inexpressive DLs: Another hindrance to the acceptance of preferential DLs is that they have only been sparsely applied to low-complexity DLs [70, 53] where the motivation for investigation is to identify fragments of DLs that will yield tractable preferential reasoning.

2.9 Overriding

One of the more recent approaches to defeasible reasoning is the work by Bonatti et al. [27] on giving a semantics for "overriding" in DLs. The approach is unique because it is targetted *specifically* at DLs. Their semantics for overriding is actually a lightweight framework for extending an arbitrary DL, say $\mathcal{DL}$, with the ability to represent and reason with so-called *defeasible inclusions* (DIs for short). The resulting extensions are called $\mathcal{DL}^N$.

From a representational point-of-view, the formalism is similar to preferential reasoning in the sense that we have two types of terminological axiom: strict subsumptions and defeasible subsumptions. In overriding the authors introduce a notion of DI (of the form $C \sqsubseteq_n D$) that is similar in style to defeasible subsumption in the Preferential context.

The intended intuitive meaning of $C \sqsubseteq_n D$ is "C's are normally also D's, unless otherwise stated". Bonatti et al. go on to define a unique notion of entailment for $\mathcal{DL}^N$ KBs and thus DIs have a *unique* intuitive meaning which closely resembles that of Reiter's defaults [161].

In the preferential case (Section 2.8), even though we had a single semantics for defeasible subsumption, the intuitive interpretation of such subsumptions can change based on which ranked model(s) we choose to define entailment. That is, its intuitive interpretation can change from a skeptical notion (typical C's are D's) in the case of Rational Closure, to a credulous notion (C's are D's unless I explicitly know otherwise) in the case of Lexicographic Closure.

Syntactically speaking, the only difference between a particular $\mathcal{DL}$ and $\mathcal{DL}^N$ is that the latter introduces a new concept name NC into the vocabulary for each $\mathcal{DL}$ concept C. The idea behind these names is to refer to the *normal* instances of a particular concept. That is, a $\mathcal{DL}^N$ interpretation $\mathcal{I}$ is a $\mathcal{DL}$ interpretation in which $\text{N}C^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ for every $\mathcal{DL}$ concept C.

Semantically speaking, satisfaction of strict subsumptions (of the form $C \sqsubseteq D$) in a $\mathcal{DL}^N$ interpretation is defined analogously to the classical DL case. Entailment is also defined analogously to the DL case. That is, the

sentences satisfied in each $\mathcal{DL}^N$ model for a KB $\mathcal{K}$, are what logically follow from $\mathcal{K}$. Bonatti et al. denote the subset of strict subsumptions in $\mathcal{K}$ with $\mathcal{S}$ and a model for $\mathcal{S} \subseteq \mathcal{K}$ is called a *pre-model* for $\mathcal{K}$.
The essential difference in $\mathcal{DL}^N$, of course, is the definition for satisfaction of a DI in a $\mathcal{DL}^N$ interpretation. Recall that all approaches to dealing with exceptions in KR formalisms must address two key issues. The first is to define *exception* in their context and define a mechanism for dealing with an exception to some default or defeasible information. The second is to define a mechanism to specify and deal with *priorities* among default sentences that conflict with each other.

In overriding these issues are dealt with by specifying the conditions under which some DI can be safely *overridden* by a normality concept. The normality concept then basically represents an exception to the DI. If we ignore priorities among DIs for the time being (i.e., assuming all DIs have equal priority), then a DI $C \sqsubseteq_n D$ is *overridden* by a normality concept $\mathrm{N}X$ in an interpretation $\mathcal{I}$ (w.r.t. a KB $\mathcal{K}$) if each pre-model $\mathcal{J}$ for $\mathcal{K}$ is s.t. either:

1. there is an $x \in \mathrm{N}X^{\mathcal{J}}$ s.t. $x \in (C \sqcap \neg D)^{\mathcal{J}}$ *or*,
2. $\mathrm{N}X^{\mathcal{J}} = \emptyset$

For example, given $\mathcal{K} = \{$ $\mathsf{EmployedStudent} \sqsubseteq \mathsf{Student}$, $\mathsf{Student} \sqsubseteq_n \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$ $\}$, one can observe that $\mathsf{Student} \sqsubseteq_n \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$ is *not* overridden by the normality concept $\mathrm{N}\mathsf{EmployedStudent}$ because we can construct a pre-model $\mathcal{J}$ of $\mathcal{K}$ s.t. for each $x \in \mathrm{N}\mathsf{EmployedStudent}^{\mathcal{J}}$, $x \in (\neg\mathsf{Student} \sqcup \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice}))^{\mathcal{J}}$ *and* $\mathrm{N}\mathsf{EmployedStudent}^{\mathcal{J}} \neq \emptyset$.

However, adding $\mathsf{EmployedStudent} \sqsubseteq \exists \mathsf{receives}.\mathsf{TaxInvoice}$ to $\mathcal{K}$ causes the DI $\mathsf{Student} \sqsubseteq_n \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$ to be overridden by $\mathrm{N}\mathsf{EmployedStudent}$ in each model for $\mathcal{K}$ because we cannot construct such a pre-model of $\mathcal{K}$.

Things become a little more involved when we introduce priorities among DIs. A priority relation $\prec$ is a strict partial order on the DIs in a KB. Although this relation can be user-specified we restrict our attention to re-

lations determined by *specificity* [27, Page 11]: $C_1 \sqsubseteq_n D_1 \prec C_2 \sqsubseteq_n D_2$ if and only if $\models C_1 \sqsubseteq C_2$ and $\not\models C_2 \sqsubseteq C_1$.
For example, let $\mathcal{K} = \{$ $\mathsf{EmployedStudent} \sqsubseteq \mathsf{Student}$, $\mathsf{Student} \sqsubseteq_n \neg(\exists \mathsf{receives.TaxInvoice})$, $\mathsf{EmployedStudent} \sqsubseteq_n \exists \mathsf{receives.TaxInvoice}$, $\mathsf{EmployedStudent} \sqcap \mathsf{Parent} \sqsubseteq \neg(\exists \mathsf{receives.TaxInvoice})$ $\}$. $\mathcal{K}$ contains two defaults where the one with antecedent $\mathsf{EmployedStudent}$ has higher priority than the one with antecedent $\mathsf{Student}$ (according to specificity).

When taking into consideration priorities among DIs, we have to introduce a third condition to characterise overriding. The full conditions are now provided in the following definition reformulated from Bonatti et al. [27, Definition 1, p12]:

Definition 24 (Overriding) *A DI $C \sqsubseteq_n D$ is* overridden *by a normality concept NX in an interpretation $\mathcal{I}$ (w.r.t. a KB $\mathcal{K}$) if each pre-model $\mathcal{J}$ for $\mathcal{K}$ is s.t. either:*

1. *there is an $x \in NX^{\mathcal{J}}$ s.t. $x \in (C \sqcap \neg D)^{\mathcal{J}}$* or,
2. *$NX^{\mathcal{J}} = \emptyset$* or,
3. *there is a DI $C_1 \sqsubseteq_n D_1 \prec C \sqsubseteq_n D$ s.t. $\{$ NW | for each $x \in NW^{\mathcal{I}}$, $x \in (\neg C_1 \sqcup D_1)^{\mathcal{I}}$ $\} \setminus \{$ NY | $C_1 \sqsubseteq_n D_1$ is overridden in NY w.r.t. $\mathcal{I}$ $\} \not\subseteq \{$ NZ | for each $x \in NZ^{\mathcal{J}}$, $x \in (\neg C_1 \sqcup D_1)^{\mathcal{J}}$ $\}$.*

Note that Definition 24 on Page 103 is recursive (condition 3 recursively refers to overriding). Intuitively, condition 3 ensures that higher priority DIs are satisfied unless they are explicitly overridden. Very informally, the set $\{$ NW | for each $x \in$ N$W^{\mathcal{I}}$, $x \in (\neg C_1 \sqcup D_1)^{\mathcal{I}}$ $\}$ represents the set of normality concepts that do *not* represent exceptions w.r.t. $C_1 \sqsubseteq_n D_1$ in $\mathcal{I}$; $\{$ NY | $C_1 \sqsubseteq_n D_1$ is overridden in NY w.r.t. $\mathcal{I}$ $\}$ represents the set of normality concepts that explicitly override $C_1 \sqsubseteq_n D_1$ in $\mathcal{I}$ and $\{$ NZ | for each $x \in$ N$Z^{\mathcal{J}}$, $x \in (\neg C_1 \sqcup D_1)^{\mathcal{J}}$ $\}$ represents the set of all normality concepts that do *not* represent exceptions w.r.t. $C_1 \sqsubseteq_n D_1$ in a pre-model $\mathcal{J}$ for $\mathcal{K}$.

For example, given the KB $\mathcal{K} = \{$ $\mathsf{EmployedStudent} \sqsubseteq \mathsf{Student}$, $\mathsf{Student} \sqsubseteq_n \neg(\exists \mathsf{receives}.\ \mathsf{TaxInvoice})$, $\mathsf{EmployedStudent} \sqsubseteq_n \exists \mathsf{receives}.\ \mathsf{TaxInvoice}$ $\}$, we can verify using Definition 24 on Page 103 that $\mathsf{NStudent}$ does not override $\mathsf{Student} \sqsubseteq_n \neg(\exists \mathsf{receives}.\ \mathsf{TaxInvoice})$ or $\mathsf{EmployedStudent} \sqsubseteq_n \exists \mathsf{receives}.\ \mathsf{TaxInvoice}$ in all $\mathcal{DL}^N$ interpretations $\mathcal{I}$. We can also verify in the same way that $\mathsf{NEmployedStudent}$ *does* override $\mathsf{Student} \sqsubseteq_n \neg(\exists \mathsf{receives}.\ \mathsf{TaxInvoice})$ in some interpretations (recall $\mathsf{EmployedStudent} \sqsubseteq_n \exists \mathsf{receives}.\mathsf{TaxInvoice} \prec \mathsf{Student} \sqsubseteq_n \neg(\exists \mathsf{receives}.\ \mathsf{TaxInvoice})$ according to specificity).

Satisfaction of a DI in a $\mathcal{DL}^N$ interpretation $\mathcal{I}$ is defined in terms of overriding. We restate the definition here [27, Definition 2, p15]:

Definition 25 (DI Satisfaction) *A DI $C \sqsubseteq_n D$ is satisfied in an interpretation $\mathcal{I}$ if each normality concept NX is either satisfied in $\mathcal{I}$ (for each $x \in NX^{\mathcal{I}}$, $x \in (\neg C \sqcup D)^{\mathcal{I}}$) or NX overrides $C \sqsubseteq_n D$ in $\mathcal{I}$.*

Given the semantic foundation, entailment for $\mathcal{DL}^N$ is defined analogously to classical entailment: a model for a $\mathcal{DL}^N$ KB $\mathcal{K}$ is a model for each of its DIs and strict subsumptions and we can derive a DI δ from $\mathcal{K}$ if and only if each model for $\mathcal{K}$ is a model for δ (written as $\mathcal{K} \mid\!\approx \delta$). Bonatti et al. [27] also give a syntactic procedure for deriving DIs from $\mathcal{DL}^N$ KBs. For pragmatic reasons, the procedure is restricted to KBs that do not contain normality concepts of the form NC because reasoning with such cases must take into account the infinitely many normality concepts in the language. The worst case complexity for performing DI inference in $\mathcal{ALC}^N$ is EXPTIME-COMPLETE (i.e., the worst case complexity of the underlying classical DL is not increased).

2.9.1 Discussion

Overriding is a fairly recent proposal for defeasible reasoning in DLs. The mechanism has been designed as a lightweight framework for extending any classical DL to be able to represent and reason with DIs. There are a number of strengths and weaknesses with this framework. Some strengths include

that it can be reduced to classical DL entailment under some restrictions on the KB and query, reasoning is tractable for some low-complexity DLs, the worst case complexity is not increased when extending a variety of DLs ranging from $\mathcal{ALC}$ to $\mathcal{SHIQ}$ and there are implementations and performance evaluations for low-complexity DLs.

Some weaknesses are that the language is perhaps too permissive which results in restrictions having to be placed on the vocabulary of KBs in order to guarantee certain desirable practical properties and to devise procedures. This also leads to the observation that the semantic characterisation of the $\mathcal{DL}^N$ framework is perhaps not as conceptually elegant as other formalisms such as default logic, circumscription and preferential reasoning.

As we shall also see in Chapter 5, overriding does not define a "rational" entailment relation (according to KLMs notion of rationality). This places some doubt over its inferential merit even though, in many representative examples in the literature, it gives back the intuitively desired inferences.

2.10 Summary and Discussion

In this chapter we have discussed the evolution of various approaches to defeasible reasoning and their contemporary application to DLs. We notice that some proposals, such as circumscription, MKNF, preferential reasoning and overriding, proposed new semantics and constructs in the object language to deal with exceptions.

Other approaches such as default and defeasible logic opted to take a rule-based (or argumentation-based) approach. MKNF did not plan to address exceptions in a direct manner, rather, it addressed a more general underlying problem which perhaps leads to exceptions in many circumstances. Its high computational complexity remains a significant drawback.

Circumscription enjoys an intuitive semantics and is well studied in the literature especially with respect to computational complexity. Unfortunately, the computational results are not encouraging and, while there are

reasonable implementations, there are not many practically viable ones for DLs. In addition, allowing full flexibility in the circumscription pattern is perhaps too permissive if we want formalisms whose consequence relations can be characterised to some degree using formal postulates.

Defeasible subsumption (preferential reasoning), defeasible rules (defeasible logic), defaults (default logic) and DIs (overriding) have a close relationship in terms of what they intuitively want to represent. Even though default and defeasible logics have various proposals for semantics, they suffer from instability in terms of agreeing upon an interpretive model-theoretic semantics. We argue that this fact contributes to some of the problems with these formalisms such as counterintuitive conclusions and odd theoretical results (e.g. general default theories do not have to have extensions [161]).

The preferential approach and overriding are much more clear than default and defeasible logic from a semantic perspective. Although, practical algorithms and implementations are just starting to emerge in the literature for the former proposals. It is clear, then, that all approaches to defeasible reasoning in DLs have their strengths and weaknesses.

In selecting a suitable formalism for practical development in this thesis, we have to agree on certain requirements for practical defeasible reasoning in DLs. We advocate the following broad requirements: (*i*) the formalism should have an intuitive representation, (*ii*) its inferential character must be "sensible", (*iii*) its complexity of reasoning must not be too much higher than the underlying classical DL it is extending and (*iv*) it must be fairly simple to implement and integrate into modern ontology editing systems.

In keeping with this philosophy, we have provisionally accepted the preferential reasoning approach as the most viable candidate to address. This is because the notion of defeasible subsumption is a representationally simple and intuitive construct. It is also the only extra expressivity added to DLs. Since preferential reasonings were historically developed first from the perspective of consequence relations, the formalisation of the inference mechanisms and semantics came later.

We believe this is a good thing. It gives a more abstract perspective of defeasible reasoning behaviour and allows one to debate on the intuition and merit of logical postulates, prior to formalising details of semantics and entailment. A major advantage to preferential reasoning is its full reduction to classical DL entailment. That is, if we extend any DL to be able to represent defeasible subsumption, we can use any sound and complete reasoning implementation for the underlying classical DL to perform defeasible inference over the added feature of defeasible subsumption.

This independence of reasoning implementation is a very positive aspect for a number of reasons and is a particularly strong advantage over most other defeasible reasoning approaches to DLs. It means that existing DL reasoners can be used off-the-shelf to perform defeasible inference and that existing tools for ontology editing and reasoning are not difficult to extend to be able to support defeasible inference (see Chapter 7).

2.11 Notation and Conventions

In this thesis we use the terms ontology and knowledge base (KB) interchangeably. We sometimes refer to the "classical counterpart" or "classical translation" of a defeasible subsumption $C \mathrel{\sqsubset\!\!\sim} D$ or, respectively, *set* of defeasible subsumptions $\{ C_1 \mathrel{\sqsubset\!\!\sim} D_1, \ldots, C_n \mathrel{\sqsubset\!\!\sim} D_n \}$. These refer to the classical versions $C \sqsubseteq D$ (respectively $\{ C_1 \sqsubseteq D_1, \ldots, C_n \sqsubseteq D_n \}$) of each construct.

Chapter 3

Requirements Analysis for Defeasible Reasoning

From a practical and empirical perspective, the state-of-the-art of defeasible reasoning research for DLs is not yet mature. One goal of this thesis is to make some headway in this area (at least for the preferential approach) to show that some formalisms are suitable for practical implementation and use in modern ontology editing systems (even in their current state).

In this chapter we address two issues with defeasible reasoning for DLs: (*i*) the lack of quantitative evidence for the need of defeasible reasoning in DLs and (*ii*) the lack of a standard for the inferential "shape" of defeasible reasoning in DLs. For (*i*), while there has been anecdotal evidence in the literature which suggests a need for the representation of defeasible information in real world ontologies, to the best of our knowledge there has been no empirical work published to confirm or refute this. Therefore, we start off by conducting a rudimentary empirical analysis of real-world ontologies to give prima facie evidence supporting that there is, in fact, a general need for representing defeasible knowledge.

Similarly, for (*ii*), there has been no investigation and standardisation of which syntactic inference rules, satisfied by classical DL inference, should be inherited by defeasible DL inference. We generalise the KLM formal

properties from the preferential reasoning setting in DLs, and motivate why these properties should be satisfied by *any* defeasible notion of subsumption (i.e. notion of default) for nonmonotonic extensions of DLs. We also discuss some semi-formal structural properties that we motivate should be satisfied by any defeasible entailment relation for DLs. We conclude with a discussion about some gaps that need to be filled.

3.1 Need for Defeasible Description Logics

One cannot deny that there has been a solid plea in the literature for the capability of representing defeasible information using formal KR languages [158, 186, 93, 176].

The general argument indicates that classical KR languages are limited when it comes to representing defeasible statements. Examples are used, predominantly from the areas of biology and biomedicine, to motivate that it would certainly be useful and desirable if KR formalisms were enriched to be able to express information that is fallible w.r.t. exceptions.

Yet, for all the purported need for defeasible representation, there seems to be no empirical evaluation of real world ontologies establishing this as an irrevocable fact.

Here we conduct a rudimentary evaluation to uncover some quantitative evidence which would indicate more clearly whether there is a need for defeasible reasoning. As one can imagine, there are a multitude of ways in which to conduct this evaluation.

One obvious approach could be centered on a kind of user study, gathering experiential information from the ontology engineer or domain expert (the users of such systems). Such a study would certainly be fruitful in giving indication of the need for defeasible reasoning but we instead take a contrasting "bottom-up" approach since it is simpler to conduct and would nevertheless give adequate preliminary indication of the need for defeasible reasoning (that can be corroborated by future user studies).

We conduct a lexical analysis of ontology documents (explained in Section 3.1.2) to identify some symptoms indicating a need for defeasible representation.

3.1.1 Dataset

A key requirement of the data that we conduct our evaluation on, is *variety* (since we wish to draw conclusions about the general need for defeasible reasoning in real world ontologies). Ideally we would like to use ontologies that are diverse in terms of domain (regardless of their structural properties).

The Manchester OWL Repository (MOWLRep) [135] is an initiative concerned with building a framework for sharing OWL ontology datasets for use in OWL empirical research. The main motivation behind the framework is to provide data that is diverse and not biased for any particular experimental application, thus hopefully making results more significant and extrapolations more accurate.

We obtain a recent snapshot of MOWLRep for our evaluation here (as well as in Chapter 6). MOWLRep itself currently consists of three main ontology datasets: the Bioportal corpus of ontologies[1], the Oxford Ontology Library (OOL)[2] and the Manchester OWL Corpus (MOWLCorp)[3].

Whereas Bioportal and OOL are generally well known and established corpora in the OWL community, MOWLCorp [132] is the culmination of a recent OWL ontology curation effort by researchers at the University of Manchester. The ontologies in this latter corpus were obtained through sophisticated web crawls and filtration techniques.

In our 2014 snapshot of MOWLRep, there are 344 ontologies in the Bioportal subset, 793 ontologies in the OOL subset and $20,996$ ontologies in the MOWLCorp subset. In terms of the average ontology metrics (such as ontology size and DL expressivity), we do not give them here because they have

[1] bioportal.bioontology.org

[2] cs.ox.ac.uk/isg/ontologies

[3] mowlrepo.cs.manchester.ac.uk/datasets/mowlcorp

little bearing on this kind of evaluation (we do, however, give such metrics in our evaluation in Chapter 6 where these characteristics become relevant).

3.1.2 Experiment Setup

The Web Ontology Language does not natively support the expression of defeasible information. What this effectively means is that any (logic-based) axioms expressed in OWL ontologies are interpreted as universally true statements about the particular application domain being described.

For the most part, this assumption is an acceptable one to make, but in certain domains it is unsuitable because there may be exceptions to the axiom. In domains where defeasible statements are more obviously required, users may still (either advertently or inadvertently) indicate that defeasible representation is needed through non-logical metadata constructs in OWL.

In this investigation we focus on *OWL Annotations* and the *string representations* of entities (concept, individual and role names) in the OWL ontology document to inform whether defeasible reasoning is either implicitly or explicitly called for.

OWL Annotations: The OWL 2 standard includes constructs in the language called OWL annotations (also called OWL annotation axioms) which can be added to ontologies to hold meta data descriptions of the ontology. Most often, these annotation axioms are used in a very similar way to computer programming comments. In other words, they are used to annotate the different parts of the ontology with natural language descriptions of the concepts, roles and axioms in the ontology, together with their utility.

OWL annotations can be attached to any entity (concept, role, individual and axiom) in the ontology. In addition, one can attach multiple annotations per entity. Figure 3.1 gives an example of annotation usage in an ontology of MOWLRep indicating a need for defeasible representation.

When we mean that a particular symptom in an ontology "indicates a need for defeasible representation", we do not necessarily mean that the ontology engineer *intended* defeasible representation from the start. What we mean is

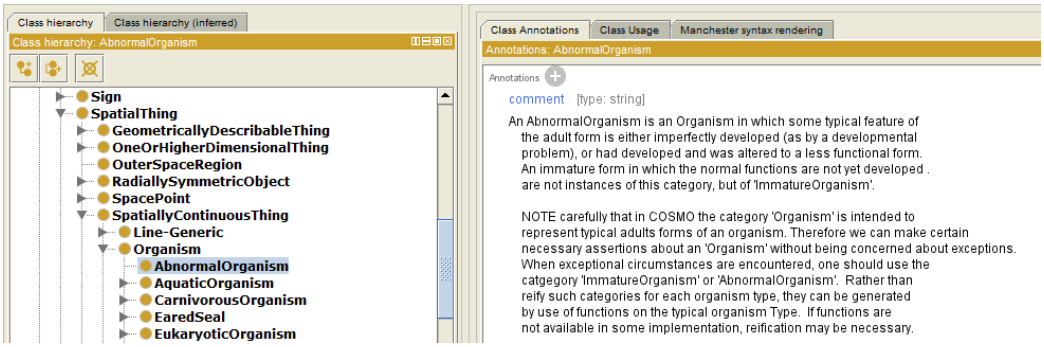


Figure 3.1: An ontology in MOWLRep loaded into the graphical ontology editor Protégé contains a class name which has an annotation indicating the need for defeasible representation.

that their chosen representation most likely is a counter-measure or "work-around" for the fact that defeasible representation is not supported in OWL.

In other words, using a suitable defeasible representation (for example defeasible subsumption) one may be able to capture the intended meaning of their representation in a more natural way. In fact, the modelling behaviour in Figure 3.1 is a common strategy to work around the lack of defeasible reasoning support. That is, the engineer creates separate classes to refer to the canonical entities of a particular type and the *exceptional* entities of that type. This strategy is actually embodied in a standardised ontology design pattern (ODP) [186, Section 5.2].

Entity String Representations: logic, as a study of deductive processes, is concerned with the "form" of propositions and not in their string representations (or the natural language meanings that we attach to these string representations). Therefore, from the point of view of deductive inference, the sentences $\mathsf{EmployedStudent} \sqsubseteq \mathsf{Student}$ are $\mathsf{C} \sqsubseteq \mathsf{D}$ are indistinguishable.

Of course, since OWL ontologies are also meant for transmission and interpretation among humans, we need to know what the symbols in our ontology represent in the real world on a lower level than their algebraic semantics. Therefore, we often use natural language terms (for the concept, role and individual names in our ontology) to refer to things we are describing.

We therefore look for clues in the lexical representation of concept and role names in the ontology to indicate where the engineer wished to indicate defeasible information. For example, encountering an axiom like $\mathsf{TypicalStudent} \sqsubseteq \neg\exists \mathsf{receives.TaxInvoice}$ may indicate that the engineer would prefer a more natural representation of the axiom using defeasible subsumption: $\mathsf{Student} \sqsubseteq \neg\exists \mathsf{receives.TaxInvoice}$.

Hence we conduct a straightforward string matching procedure on each class, role and individual name (as well as annotations) in the ontologies of MOWLRep. We specifically search for string expressions containing terms that are related in natural language meaning to "exceptions", "typicality", "abnormality" etc. We make use of both synonym and antonym terms to get adequate coverage of the cases. In many cases we only need to search for the "root" word e.g. "normal" and we automatically detect cases where the words "normally" and "abnormal" occur.

The following list is exhaustive w.r.t. the root expressions that we search for in the terms of the ontologies:

- "normal", "abnormal".
- "typical", "atypical", "prototypical".
- "standard", "non-standard", "nonstandard".
- "regular", "irregular".
- "except" (includes "exception" and "exceptional").
- "usual", "unusual".
- "conventional", "unconventional".
- "canonical", "non-canonical", "noncanonical".
- "anomalous", "anomaly".

When searching the concept, role and individual names in the ontology, we want to ensure that we detect accurate cases such as "normalStudent" and "typicalMammalianCell". We also want to eliminate "false" hits for e.g. "normals" and "standards" (referring to vector normals in linear algebra and relative conventions respectively). Therefore, we add such cases manually as exceptions to ignore in our search list of expressions.

We also ensure that we search only for cases where our root expressions are not *suffixes* in the term. We conjecture that it is far more likely to be a false hit when the root expression occurs as a suffix of the term. This is because ontology engineers often adopt the convention of using nouns as concept names. Since our root expressions are all adjectives, they would most likely *precede* a noun in natural language grammar.

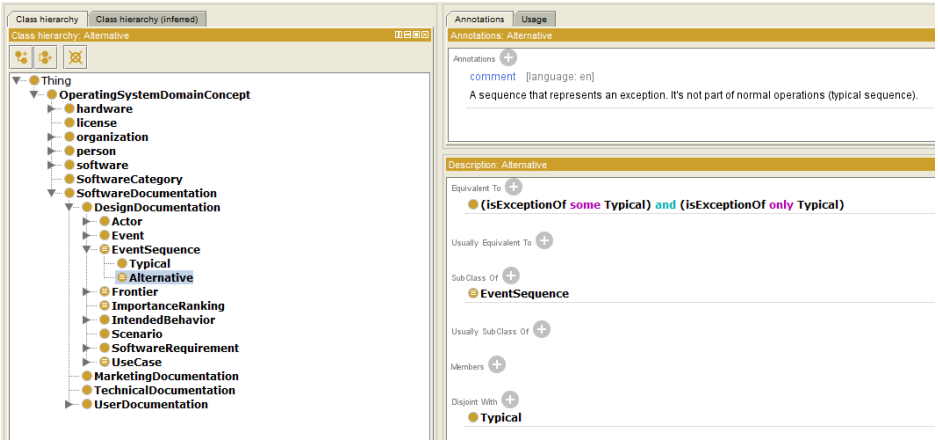
This strategy eliminates false hits such as "highStandard" and "longNormal". In our evaluation we found that the root expressions usually occur as adverbs in role names (for e.g. "usuallyComposedOf" and "typicallyLocatedIn"). Figure 3.2 gives some examples of names in ontologies indicating need of defeasible representation.

To summarise our methodology, we search for our root expressions in each ontology document in MOWLRep. If an ontology contains a hit (an entity contains the given expression either in a name or annotation) then we regard the ontology as a successful hit. We count the number of ontologies that are successful hits in MOWLRep.

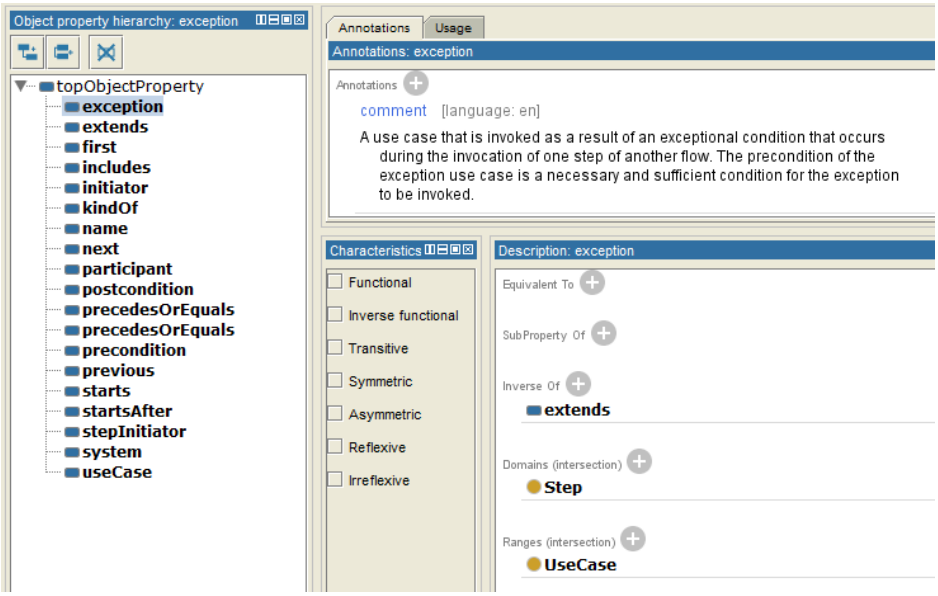
3.1.3 Results and Discussion

Since our evaluation is of a rudimentary nature, the results can be accurately summarised in the single table of Figure 3.3.

It is interesting to note the considerable percentage of ontologies of BioPortal and OOL that contain hits (22% and 27% respectively). Also, it is interesting that the ontologies of BioPortal do not contain any hits whatsoever in annotations. The numbers, we conjecture, suggest that there is a significant need for defeasible representation in OWL ontologies. We have to



(a) An ontology about operating systems in MOWLRep indicating the need for defeasible constructs.



(b) An ontology describing aspects about use cases containing a role name which indicates need of defeasible expression.

Figure 3.2: Ontologies in MOWLRep loaded into the graphical ontology editor Protégé containing names indicating the need for defeasible representation.

point out to the reader that there are factors in our evaluation which may incorrectly inflate this significance. But there are also factors which may incorrectly deflate this claimed significance.

Corpus	Total Ontologies	Ontology Hits				
		String	Annotations	Both	Total	Percentage (%)
Bioportal	344	75	0	0	75	21.8
OOL	793	181	16	19	216	27.2
MOWLCorp	20,996	1,004	275	39	1,318	6.3

Figure 3.3: Results for evaluation: "String" column indicates ontologies which contained hits only in string names, "Annotations" column indicates ontologies which contained hits only in annotations, "Both" column indicates ontologies which contained hits both in string names and annotations.

A successful hit in our evaluation does not state unequivocally that there is a definite need for defeasible representation in that ontology. There are perhaps some hits for which this is not the case. However, we claim that in most cases the hits connote a need for defeasible reasoning support (although we defer a deeper analysis giving evidence for this). On the other hand there are factors in the data which could understate the need for defeasible reasoning. The most important one is that contemporary ontology engineers may have learned to avoid modelling choices leading to symptoms displaying a need for defeasible reasoning.

That is, because defeasible reasoning support is not available, work arounds are created to model exceptions in alternative ways that do not require defeasible representation. In fact, conventions such as abiding by monotonic knowledge representation [24, Page 9], and design patterns to model exceptions, have been developed to address this [186, Section 5.2].

Therefore, even though our evaluation is by no means a comprehensive indicator that defeasible reasoning support is needed in OWL, the results add some numerical value to the anecdotal examples in the literature motivating the need for defeasible reasoning. Hence, in summary, we believe that the numbers of our evaluation, together with the pleas in the literature, are sufficient grounds from which to claim that the need for defeasible reasoning in OWL is significant.

3.2 Inferential Character

At their core, classical logics all have a universally agreed upon character to them: usually they consist of a language (a set of symbols) that may be combined using specified syntactical and grammatical rules to form propositions (logical sentences), together with some semantic machinery for interpreting the meaning of these sentences in some model theory.

One unifying characteristic of classical logics with a model-theoretic semantics is that entailment from a set of propositions (logical theory) is defined as what holds in *all* models of the theory. However, as we know, classical logic did not, historically speaking, start out with fully formed model theories.

Rather, there was simply a syntax for representing propositions. Entailment was defined by formulated *inference rules* (born from the furnace of hotly contested philosophical debate). Of course, Alfred Tarski then paved the way for model theories with his definition for the logical truth of a proposition relative to its satisfaction in a relevant "interpretation".

Therefore, if we are concerned with what propositions follow from theories expressed in classical logics, then we can study the question both from the perspective of their model theories, and from the perspective of the properties (inference rules) that these model theories actually obey.

Now there is an argument that, in the case of defeasible extensions to classical formalisms, it is desirable to retain as many of these properties as possible. I.e., the argument goes that the inferential character of defeasible (or nonmonotonic) formalisms should only deviate from classical formalisms where necessary in order to effectively take into account the exceptions.

KLM discussed which properties to retain for nonmonotonic inference (specifically in the context of propositional logic) and agreed upon the ones we recapped in Section 2.8.1. Reformulating these properties for the DL case is a straightforward affair [42, Definition 2] but explicit motivation for *why* they should be satisfied in this setting is still missing in the literature.

We address this concisely in the following subsections by using real world examples to indicate *why* it is sensible for defeasible reasoning mechanisms

in DLs to satisfy these properties.

3.2.1 Formal Properties

We recap here the formal properties that should be satisfied by rational nonmonotonic consequence relations interpreted in the context of DLs [42, Definition 2]. While it has been argued by KLM that these properties should be satisfied by nonmonotonic extensions of propositional logic, it has not been sufficiently argued in the DL case.

Here we motivate with examples that it is sensible to require that these properties be satisfied for nonmonotonic extensions of DLs as well. It is important to note that we do not claim that these are the *only* properties that should be satisfied by nonmonotonic extensions of DLs. Rather, we recommend that these are the *minimal* properties that should be satisfied in order to be called rational.

Of course, motivation of abstract properties of this nature in logic is a philosophical matter. Hence, there are likely to be various competing perspectives to our own about these properties. Our general view is that we are confident in the integrity of these properties as minimal requirements of a rational nonmonotonic reasoning behaviour, but, at the same time, we are open to possible *variants* of these properties. That is, if there are any disagreements with these properties, it is more likely that they could be resolved with fairly minor modifications to the properties, rather than entirely omitting any of them as requirements of rational nonmonotonic inference.

The properties we are discussing pertain to a notion of defeasible consequence with respect to DLs. "Consequence" here can be interpreted on both the object and entailment levels. As we know, in classical DLs, subsumption defines the main notion of consequence (between DL concepts) on the object level. Therefore, the properties we are going to present essentially define a class of notions of (defeasible) subsumption (i.e., exactly those which satisfy the properties themselves).

We define a defeasible conditional relation $\rightsquigarrow$ (a placeholder for our var-

ious possible notions of subsumption) as a binary relation on the concept language for $\mathcal{ALC}$. We recommend that any $\rightsquigarrow$ should satisfy the following seven properties (the KLM properties) where C, D and E represent $\mathcal{ALC}$ concepts:

$$\text{(Ref) } C \rightsquigarrow C \qquad \text{(LLE) } \frac{\models C \equiv D,\ C \rightsquigarrow E}{D \rightsquigarrow E}$$

$$\text{(And) } \frac{C \rightsquigarrow D,\ C \rightsquigarrow E}{C \rightsquigarrow D \sqcap E} \qquad \text{(Or) } \frac{C \rightsquigarrow E,\ D \rightsquigarrow E}{C \sqcup D \rightsquigarrow E}$$

$$\text{(RW) } \frac{C \rightsquigarrow D,\ \models D \sqsubseteq E}{C \rightsquigarrow E} \qquad \text{(CM) } \frac{C \rightsquigarrow D,\ C \rightsquigarrow E}{C \sqcap D \rightsquigarrow E}$$

$$\text{(RM) } \frac{C \rightsquigarrow E,\ C \not\rightsquigarrow \neg D}{C \sqcap D \rightsquigarrow E}$$

One possible reading of the sentence $C \rightsquigarrow D$ is "D is plausibly derivable from C". In this sense, $C \rightsquigarrow D$ can be thought of as a general form for default statements in DLs. We can either directly substitute or interpret default statements from specific defeasible DL formalisms in this more general default form to evaluate the particular formalism against the above postulates. We discuss the merit of *any* such notion of defeasible subsumption abiding by these properties below, together with some motivating examples for the context of DLs.

Reflexivity (Ref): The reflexivity property is quite uncontentious and intuitive. Since $\rightsquigarrow$ is considered here as a relation between $\mathcal{ALC}$ concepts, we have to consider the semantics of concepts when interpreting this property. Obviously, we know that classical subsumption respects (Ref) because a set is always a (nonstrict) subset of itself. However, classical subsumption is monotonic and is therefore not a relevant interpretation of $\rightsquigarrow$ for our purposes. $\rightsquigarrow$ has to be interpreted as a *defeasible* notion of subsumption.

An important agreement to make is our natural language reading of $\rightsquigarrow$, i.e., the intuitive notion of defeasible consequence that this relation is intended to capture on the object level. Given a conditional sentence $C \rightsquigarrow D$ where C and D are $\mathcal{ALC}$ concepts we choose the reading "usually, objects

having the property C also have the property D". It is clear, given this reading, that (Ref) is a sensible constraint to satisfy for defeasible inference in DLs. In fact, "usually, objects having the property C also have the property C" is a tautological statement[4] no matter our interpretation of *usually*.
Left Logical Equivalence (LLE): Left logical equivalence holds that, if two sets are logically indistinguishable (contain the same elements), then either set will share the same "default" properties.

For example, consider the concepts Student and Person $\sqcap$ $\exists$enrolledIn. EducationalInstitution representing the set of all students and the set of all persons that are enrolled at some educational institution, respectively.

If the aforementioned concepts actually refer to the same set of objects from the domain of discourse, and it is further known that students usually don't pay taxes, then nothing is preventing us from concluding that persons who are enrolled at educational institutions also usually don't pay taxes.

This is, of course, all predicated on the fact that we accept the constraint $\models$ Student $\equiv$ Person $\sqcap$ $\exists$enrolledIn. EducationalInstitution. That is, exceptions are not permitted to this equivalence.

It then makes sense to attribute the properties typically associated with Student, to Person $\sqcap$ $\exists$enrolledIn.EducationalInstitution as well, since these are just distinct names for describing the same set of entities in the domain.
(And): If it is known that Swedes are usually tall, and it is also known that Swedes are usually blonde, (And) endorses the conclusion that Swedes are usually *both* tall and blonde.

A description more reminiscent of the concept semantics of DLs is that: if we can *usually* attribute the property of tallness and blondeness to the *same* group of entities in the domain, then there appears to be no rational reason to prevent both properties to be attributed *simultaneously* to the group.

In other words, no matter our formalisation of "usually", we talk here about two typical (although independent) properties of the *same* group (i.e., Swedes), and if we are willing to defeasibly attribute these properties to

[4]Note the hint at supraclassicality in the statement.

Swedes *independently*, then it makes sense to attribute them *simultaneously* to this group.

(Or): If it is known that Swedes are usually tall, and it is also known independently that basketball players are usually tall, (Or) endorses that a group of entities consisting of either Swedes or basketball players (or both) are usually tall as well.

If we interpret this rule in the setting of propositional logic, it implies that if independent premises lead to the same defeasible conclusion then their disjunction is still sufficient to entail that conclusion. Essentially this property defines a compelling condition under which a specific generalisation or inductive inference can be made.

In other words, if I can attribute a default property to a number of (possibly disjoint) sets of these entities, then I can attribute the default property to the union of these sets of entities. This is a fairly straightforward and intuitive property to retain from classical logic.

Right weakening (RW): If it is known that mechanics are usually male, and that males *infallibly* have a Y-chromosome, then (RW) will endorse the conclusion that mechanics usually have a Y-chromosome. In other words, it makes sense to attribute to the typical properties of a group, all its *universally true* meta-properties.

An alternative description is that if I am ready to attach a plausible attribute to a group of entities, then I should also be ready to attach to these entities all *strict* (universally true) conclusions from this plausible attribute.

(RW) actually defines a property which is similar to transitivity for $\rightsquigarrow$ even though it is not the natural translation of classical transitivity. A natural translation of the classical transitivity property for $\rightsquigarrow$ is:

$$\text{(Trans)}\ \frac{C \rightsquigarrow D,\ D \rightsquigarrow E}{C \rightsquigarrow E}$$

However, endorsement of (Trans) is perilous because, in the propositional case, KLM have showed that, in the presence of (Ref), (LLE), (RW), (And), (Or), (CM), (Trans) is equivalent to monotonicity [111] and, by application

of these aforementioned properties, it can easily be shown that this result generalises to the DL case as well.

Of course, transitivity would not be a useful property to endorse for defeasible consequence because, for example, if high school dropouts are usually adults, and adults are usually employed, one should not necessarily conclude that high school dropouts are usually employed [165, Section 2]. In one particular interpretation of *usually*, this example alludes to the problem with accepting (Trans). That is, just because I believe that high school dropouts are usually adults, it does not mean that I believe that they are *typical* adults. It may be that they are *abnormal* sorts of adults (w.r.t. to some attribute). Therefore, the next statement: adults are usually employed, may lose the transitive link with the first statement because we may be implicitly referring to *typical* adults in the second statement.

Cautious Monotonicity (CM): The last two properties we discuss are perhaps the most interesting and significant ones when it comes to defining defeasible inference mechanisms. This is because these properties simulate "weaker" forms of monotonicity.

In other words, the idea here is that just because we do not want defeasible inference mechanisms to be monotonic in general (indeed that is the very property we want to avoid in general), it shouldn't mean that we should not abide by some "cumulative reasoning" properties.

Classical monotonicity says that: "in *all* circumstances, all facts stated before still have to hold and peacefully coexist with new facts as they come to light". In other words, in this case reasoning is *always* cumulative.

However, just because we do not want cumulativity to hold *all* of the time, it does not mean that it is not a reasonable inferential behaviour in *some* circumstances. Cautious monotonicity (and later Rational monotonicity) are proposals for isolated circumstances in which monotonic (or cumulative) reasoning behaviour is perhaps still sensible to inform defeasible consequence.

While $C \rightsquigarrow E$ is sufficient grounds for (M) to be able to endorse $C \sqcap D \rightsquigarrow E$ (for any D), (CM) is more "cautious" by only allowing D's that can be

plausibly derived from C. For example, one may accept that cakes are usually delicious, and whereas this is sufficient information for (M) to conclude that cakes made with rotten eggs are also usually delicious, (CM) will only allow one to draw this conclusion if rotten eggs are a typical property of cakes (which, as we know, they are not).

To give a positive example for (CM): if we accept that cakes are usually delicious, then (CM) will accept that cakes containing sugar are also usually delicious because the property of containing sugar can be plausibly attributed to cakes themselves (i.e., cakes usually contain sugar).

Rational Monotonicity (RM): Rational monotonicity is inferentially stronger (more permissive) than (CM). That is, (RM) poses weaker premises than (CM) from which one can make the same cumulative inference. It holds that given $C \rightsquigarrow E$ (E is plausibly derivable from C), as long as one *cannot* plausibly derive the *negation* of D from C, then adding D to our knowledge does not affect our plausibility of deriving E.

To use the cake example, if we know that cakes are usually delicious, and we cannot say that cakes usually don't contain salt (there may indeed be recipes of cakes that contain a smidge of salt), then it is reasonable to conclude that cakes containing salt are usually delicious.

Essentially, there is no indication in our premises that the ingredient of salt is "abnormal" in cakes, and therefore, its presence in cakes shouldn't affect what plausible inferences we can draw about cakes in general.

It is worth mentioning that (RM) is slightly different in form to the aforementioned properties in that it is a "negative" property. That is, it talks about things that we *don't* know in addition to what we know in its premises, whereas the other properties only discuss what we *do* know.

This latter aspect of (RM) becomes more significant in Section 3.2.2 when we discuss some semi-formal properties of nonmonotonic inference.

In conclusion, w.r.t. the properties discussed above, KLM recommend that they be satisfied by any nonmonotonic $\rightsquigarrow$ that is to be considered "rational" and we agree with this view. Again, this is not to say that we believe

these are the *only* properties that such a relation should obey, but rather that these are the minimal properties to respect.

Indeed our main argument, from the examples we have given, is that there seems to be no reason *not* to obey any of these properties. Hence, if any nonmonotonic consequence relation for DLs does not obey one or more of these properties, it is likely to diminish the logical merit of this relation (at least somewhat).

It is also worth mentioning that some of the properties mentioned in this section are actually *derived* from other more elementary properties which are documented in the literature [111]. In other words, satisfaction of certain combinations of these elementary properties may imply satisfaction of one of the properties discussed here.

3.2.2 Semi-Formal Properties

Although we have motivated that the logical properties in Section 3.2.1 should be satisfied by nonmonotonic extensions for DLs, it may make sense for these extensions to satisfy additional logical properties as well. There has been some interesting work (in the setting of propositional logic) to identify some of these properties and also relate them to the existing ones [26].

Yet, beyond the realm of *formal* properties, there appear to be other "inferential rules" that seem very reasonable to obey but are difficult to describe in a formal sense. Such rules can, however, be explained intuitively. Daniel Lehmann discussed five such rules in his exposition of Lexicographic Closure for a defeasible extension of propositional logic [116].

We now discuss these properties in the context of DLs and motivate their applicability in defining rational defeasible entailment regimes.

Presumption of typicality: Given concepts C, D, E and the premise $C \rightsquigarrow E$, supposing that $C \rightsquigarrow \neg D$ is *not* endorsed, we have to accept $C \sqcap D \rightsquigarrow E$ according to (RM). However, if $C \rightsquigarrow \neg D$ *is* endorsed, we can either accept or *not* accept $C \sqcap D \rightsquigarrow E$. Either option is consistent with (RM).

Presumption of typicality prefers to endorse cumulativity unless there is

compelling evidence *not* to. That is, we prefer to endorse $C \sqcap D \rightsquigarrow E$. For DLs this means that we should be ready to treat any subclass of entities of some more general class as *typical* members of this class, unless there is explicit evidence to the contrary.

For example, suppose we know of a student named *John*, and we also know that students usually don't pay taxes. Assuming we have no other information, we must endorse that it is plausible that *John* does not pay taxes. If we later find out that John is employed then we should still endorse that *John* plausibly does not pay taxes, because we do not know anything about employed students which differentiates them from typical students.

In other words, given our information, there is nothing to suggest that employed students are exceptional. This is the idea behind presumption of typicality: presume that we are in a typical situation *unless* there is evidence to the contrary. An example of such evidence for our example is, of course, finding out that employed students are usually obliged to pay taxes. In such a case we would realise that employed students are exceptional w.r.t. typical students (because they conflict w.r.t. the property of paying taxes).

This is clearly a very reasonable property to endorse for defeasible inference in a variety of situations. Not accepting this property would define a more cautious or skeptical reasoning behaviour which may be desirable in certain settings. However, if a more adventurous reasoning behaviour is required, then presumption of typicality is a sensible rule to obey.

Presumption of independence: Presumption of independence takes a step further from presumption of typicality. If I know that students usually don't pay taxes ($\mathsf{Student} \rightsquigarrow \neg(\exists \mathsf{receives.TaxInvoice})$) *but* I also know that students are usually *not* married ($\mathsf{Student} \rightsquigarrow \neg(\exists \mathsf{marriedTo.Person})$), then should I accept the conclusion that students who are married usually *don't* pay taxes ($\mathsf{Student} \sqcap \exists \mathsf{marriedTo.Person} \rightsquigarrow (\neg\exists \mathsf{receives.TaxInvoice})$)?

One argument is that we shouldn't accept this conclusion because if students are usually not married, it means that married students refer to an atypical subclass of students, and it therefore seems reasonable in some sense

to be cautious in transferring the property of not paying taxes to these students. However, presumption of independence endorses the conclusion that married students usually don't pay taxes. The argument for this is obviously that matrimonial matters and tax paying status are independent of each other (from an intuitive perspective).

Yet another reading of this property can be: if no relationship is specified between two classes of entities (representing default properties) then one can presume that there is none. Therefore, just because a student is married, if there is no evidence to suggest it, one should not assume that it affects their tax paying status.

Priority to typicality: Sometimes the conclusions of presumption of independence and presumption of typicality conflict with each other. The question arises: which rule should we prefer in such circumstances?

Daniel Lehmann argues in the propositional case that one should give preference to presumption of typicality. We agree with this view in the DL case as well and motivate with an example.

Suppose we know that students generally don't pay taxes. Suppose we also know that students who are employed generally *do* pay taxes. Presumption of typicality endorses the conclusion that students who are employed *and* have children generally *do* pay taxes, because there is no other information forcing us to derive that students who are employed generally don't have children (i.e., that employed students who have children are abnormal employed students).

However, applying presumption of independence, we can derive that students who are employed and have children generally *do* pay taxes (same as above), *as well as*, the conclusion that students who are employed and have children generally do *not* pay taxes.

The former conclusion is derived because we presume that the properties of having children and tax paying status are independent. The latter conclusion is derived because we presume that being employed *and* having children is independent of tax paying status. Since it is clear that accepting both the

above conclusions is contradictory, we have to pick just one.

We therefore look to presumption of typicality to resolve the conflict and accept whatever conclusion that this rule endorses. Presumption of typicality cannot lead to the dichotomous conclusions that presumption of independence can sometimes lead to, because the rule purely inherits the properties of the typical members of the given antecedent class (and these properties themselves will be consistent w.r.t. each other).

Respect for specificity: One may usually attribute some property to a general class of objects, but, if some property of a *subclass* of these objects clashes with the property attributed to the general class, then what should we do? Should we prefer to "override" the property of the general class (i.e., prefer information that is more specific) or should we prefer to retain the property of the general class (i.e., reject the more specific information)?

Students usually don't pay taxes, employed students are more specific types of students and they usually *do* pay taxes. Even more specific than employed students are those employed students who have children (they may receive tax exemptions based on their situation). Note the violation of monotonicity in this reasoning behaviour.

In other words, if I wish to draw some conclusion about the tax paying status of a particular student, I should prefer to use the maximum knowledge (the most specific knowledge) available about this student in order to inform my conclusion. In this connection, we humans tend to regard inferences made using more specific information about the premise, to be more sound than inferences made using less (more general) information about the premise.

This is certainly rational behaviour and harks back to what we said about incomplete knowledge in Section 2.3. That is, having incomplete knowledge about a situation is often the very circumstance forcing us to make defeasible statements in the first place. As the saying goes, "knowledge is power", and the more (reliable) knowledge we have about a situation, the more we should trust the inferences drawn about it.

In other words, more specific information about a situation is *more* infor-

mation about it and would thus lead to more sound inferences about it.
Avoidance of junk: This property simply says that we should endorse the least number of conclusions that are induced by the formal and semi-formal properties on a starting set of premises. In other words, among all the consequence relations satisfying all the properties, the ones we should choose are those that are smallest (according to set inclusion).

3.3 Discussion

In this chapter we started off by giving quantitative evidence suggesting that the need for defeasible representation in DL-based ontologies is significant. We also motivated the main inferential properties that one should retain from classical logics when moving to the defeasible case (for DLs).

While we argued for the minimal properties that defeasible mechanisms should satisfy, there remains work to be done in studying properties that are related to the aforementioned ones and comparing them in terms of inferential strength a la Bezzazi, Makinson and Pérez [26].

There may also be novel semi-formal properties that one could introduce to inform defeasible consequence in DLs. In this chapter we focused purely on generalising Lehmann's arguments for his five properties, in the propositional setting, to DLs. Another important issue is the *formalisation* of Lehmann's semi-formal properties. We hold that if such properties could not be formalised in some way (even under certain restrictions or assumptions), then their usefulness in defining sensible defeasible entailment is very limited because we cannot definitively prove if a formalism actually satisfies them.

Investigations are also needed to establish performance requirements of defeasible reasoning in DLs. Of course, once defeasible formalisms are adopted and integrated into ontology editing tools, there will be more data from which to extrapolate projections about this required performance.

Any comments thus far on performance can only be speculative in nature, because there can be no strong indication of *how much* longer one should be

willing to wait for defeasible inference over classical inference.

Just as classical DLs have a series of popular standard and non-standard reasoning tasks which users most often utilise to gain insight into their ontologies and attached data, defeasible DLs are yet to establish the most pertinent standard and non-standard reasoning tasks for this setting.

We foresee that it is not just a straightforward reformulation of analogous reasoning tasks for the defeasible context. Principles such as concept unsatisfiability and ontology inconsistency would invariably have different meanings in the defeasible context. Non-standard tasks such as classification would also need to be reformulated and redefined for the defeasible case (if these are actually interesting reasoning problems in the new context).

Finally, there may be *other* novel reasoning problems that are unique to the defeasible case. For example, one may want to calculate the list of exceptions (exceptional entities) in an ontology of defeasible statements. One may also require to compute the hierarchy of exceptions (the exceptions-to-exceptions taxonomy) of the ontology.

Such considerations need to be investigated and formalised, if need be, in order to pave the way for integration and development of defeasible representation and reasoning services for ontology editing tools.

Chapter 4

Algorithms for Defeasible Reasoning

This chapter introduces algorithms for computing various notions of defeasible entailment within the context of preferential DLs. Each form of entailment mentioned herein has its respective strengths and weaknesses, and we discuss these when we present each proposal. A substantial advantage to our algorithms are that they reduce to a series of classical DL entailment checks, and hence, we start off this chapter by explaining how this reduction works and why it is possible. We then establish an algorithm for deciding if a defeasible subsumption is in the Rational Closure of a defeasible KB.

Section 2.8.2 explained that Rational Closure is the only known *rational* consequence relation that exhibits both *syntax-independence* and a clear model-theoretic definition. The remainder of the algorithms in this chapter compute *syntax-dependent* variants of Rational Closure, which serve as alternatives to its cautious reasoning paradigm. While these algorithms all follow the basic structure of the Rational Closure algorithm, they all have significant differences from this basic structure. For each algorithm, we demonstrate the computational complexity, termination, soundness and completeness.

4.1 Exceptionality to Classical Entailment

The central notion of reasoning with DLs that express defeasible subsumption is concept exceptionality (Definition 14 on Page 91). Firstly, we recall that concept exceptionality induces an a priori ordering on the defeasible subsumptions in a defeasible KB. This ordering is determined by the degree of exceptionality of the defeasible subsumptions in the KB, and Definitions 17 and 18 allude to a procedure for computing this ordering. We will present such a procedure later in the chapter and the reader will notice that it forms the core of all our presented algorithms for deciding defeasible entailment. For now, we prove some foundational properties of ranked models which are useful to demonstrate how verification of concept exceptionality can be accomplished using classical DL entailment.

4.1.1 Disjoint Union of Ranked Interpretations

In this section we generalise the definition of *disjoint union of interpretations* [109, Lemma 11], [10, Theorem 5.12], [168, Section 6.7] from classical DLs to preferential DLs. We show that, in the preferential case, we have two notions of disjoint union (DU for short) of *ranked* interpretations. We also prove some properties of these constructions which are useful in the sequel. We start off by recalling the definition of disjoint union of interpretations in the classical DL case.

Definition 26 (DU of Interpretations) *Let $\mathcal{K}$ be a classical DL TBox and $\mathscr{I} = \{\mathcal{I}_1, ..., \mathcal{I}_n\}$ a finite set of interpretations s.t. $\Delta^{\mathcal{I}_1}, \ldots, \Delta^{\mathcal{I}_n}$ are pairwise disjoint. Then, the* disjoint union *of each $\mathcal{I}_i$ in $\mathscr{I}$ (denoted by $\mathcal{I} = \biguplus \mathscr{I}$) is defined as $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ where $\Delta^{\mathcal{I}} = \bigcup_{i=1}^{n} \Delta^{\mathcal{I}_i}$ and $\cdot^{\mathcal{I}}$ is s.t. $A^{\mathcal{I}} = \bigcup_{i=1}^{n} A^{\mathcal{I}_i}$ and $R^{\mathcal{I}} = \bigcup_{i=1}^{n} R^{\mathcal{I}_i}$ for any concept name A (resp. role name R).*

It turns out that, for $\mathcal{ALC}$, the disjoint union of an arbitrary set of *models* for a TBox is also a *model* for the TBox. This is known as the *disjoint union*

model property and is a well known result in the literature [109, Lemma 11], [128], [10, Theorem 5.12], [168, Section 6.7], although we restate it here.

Lemma 1 (DU of Models are Models) *Given an $\mathcal{ALC}$ TBox $\mathcal{K}$, a finite set of* models *$\mathscr{I}$ for $\mathcal{K}$ (all of whose domains are pairwise disjoint) and $\mathcal{I} = \biguplus \mathscr{I}$, then $\mathcal{I} \Vdash \mathcal{K}$.*

A direct proof for Lemma 1 on Page 132 is elusive in the DL literature. However, it is known that DLs have a close correspondence with *Modal Logics* [172, 173, 58] and there is an indirect proof in this setting [59, Lemma 4.2] that can be adapted for Lemma 1.

Now, in the preferential case, we wish to define a similar notion of disjoint union but for *ranked* interpretations. Because of the stratification of the domain elements in ranked interpretations, one can define two notions of disjoint union: *horizontal* and *vertical* disjoint union of ranked interpretations. Loosely speaking, a horizontal disjoint union of ranked interpretations places the candidate ranked interpretations "side-by-side" next to each other when taking their union, while a vertical disjoint union of ranked interpretations "piles" the ranked interpretations one-by-one on top of each other.

Definition 27 (Horizontal DU of Ranked Interpretations) *Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and a finite set of ranked interpretations $\mathscr{R} = \{\mathcal{R}_1, ..., \mathcal{R}_n\}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. $\Delta^{\mathcal{R}_1}, \ldots, \Delta^{\mathcal{R}_n}$ are pairwise disjoint, the* horizontal disjoint union *of each ranked interpretation in $\mathscr{R}$ (denoted by $\mathcal{R} = \boxplus \mathscr{R}$) is defined as $\mathcal{R} = \langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}}, \prec_{\mathcal{R}} \rangle$ where $\Delta^{\mathcal{R}} = \bigcup_{i=1}^{n} \Delta^{\mathcal{R}_i}$, $\cdot^{\mathcal{R}}$ is defined s.t. $A^{\mathcal{R}} = \bigcup_{i=1}^{n} A^{\mathcal{R}_i}$ and $R^{\mathcal{R}} = \bigcup_{i=1}^{n} R^{\mathcal{R}_i}$ for any concept name A (resp. role name R), and $\prec_{\mathcal{R}}$ is defined s.t. the rank of each element $x \in \Delta^{\mathcal{R}}$ is the same as its rank in its "original" ranked interpretation. That is, for each $x \in \Delta^{\mathcal{R}}$, $rk_{\mathcal{R}}(x) = rk_{\mathcal{R}_i}(x)$ if and only if $x \in \Delta^{\mathcal{R}_i}$ for some $1 \leq i \leq n$.*

It is clear that Definition 27 on Page 132 closely mirrors Definition 26 on Page 131 except that, in addition, we have to handle the ordering component

of ranked interpretations in Definition 27. We prove that the disjoint union model property extends to the horizontal disjoint union of ranked models as well.

Lemma 2 (Horizontal DU of Ranked Models are Ranked Models) *Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, a finite set of* ranked models *$\mathscr{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ (all of whose domains are pairwise disjoint) and $\mathcal{R} = \boxplus \mathscr{R}$, then $\mathcal{R} \Vdash \langle \mathcal{T}, \mathcal{D} \rangle$.*

Proof: The proof is quite straightforward and would be analogous to a proof (in the DL case) for Lemma 1 on Page 132. Notice from Definition 27 on Page 132 that we are not altering the interpretation of the elements in the combined ranked interpretation (i.e., they inherit their interpretation from their originating ranked model). In addition, we notice that for each $C \sqsubset\!\!\sim D \in \mathcal{D}$, $rk_{\mathcal{R}}(C) = min\{rk_{\mathcal{R}_1}(C), ..., rk_{\mathcal{R}_n}(C)\}$. In other words, the rank of the left hand side (LHS) concept C of each defeasible subsumption (in the combined ranked interpretation) will be the minimal value from the ranks of C in each constituent ranked model. By definition of ranked model, the minimal C objects in this constituent model will have to satisfy D and therefore will satisfy D in the combined interpretation as well. See Figure 4.1 for a graphical representation of the proof strategy. □

We can also define a notion of *vertical* disjoint union of ranked interpretations. Informally, we place ranked interpretations one-by-one "on top of each other". These kinds of constructions differ significantly from horizontal disjoint unions of ranked interpretations because we have to redefine the ranks of elements of the domain (we cannot preserve the rank of elements in their originating interpretations as we did previously). Because the *sequence* in which we combine the ranked interpretations will affect the rank of the elements in the domain, we assume an *ordering* on the given ranked interpretations. That is, we have to specify the order in which we pile the interpretations on top of each other.

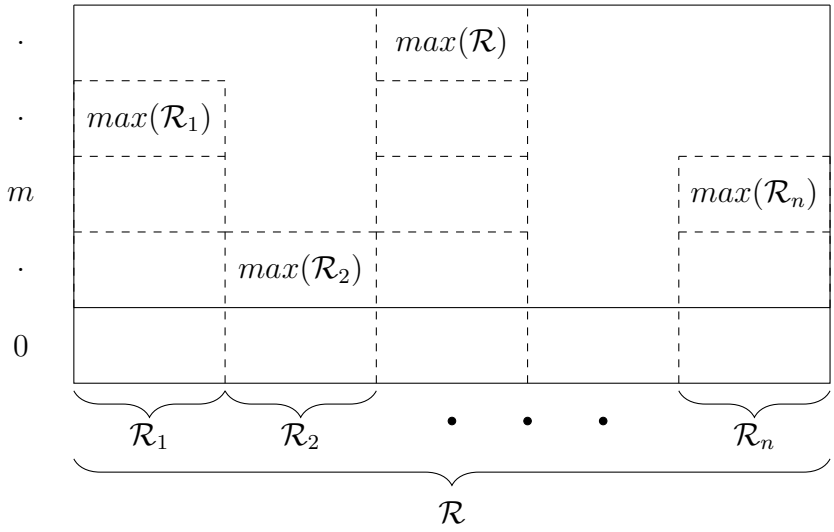


Figure 4.1: Graphical illustration of constructing the horizontal disjoint union of ranked interpretations $\mathcal{R}_1$, ..., $\mathcal{R}_n$ to form $\mathcal{R}$.

Definition 28 (Vertical DU of Ranked Interpretations) *We let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a defeasible KB and $\mathscr{R} = \{\mathcal{R}_1, ..., \mathcal{R}_n\}$ a finite* ordered *set of ranked interpretations s.t. $\Delta^{\mathcal{R}_1}, \ldots, \Delta^{\mathcal{R}_n}$ are pairwise disjoint, the* vertical disjoint union *of each ranked interpretation in $\mathscr{R}$ (denoted by $\mathcal{R} = \boxminus \mathscr{R}$ or $\mathcal{R} = \mathcal{R}_1 \boxminus ... \boxminus \mathcal{R}_n$) is defined as $\mathcal{R} = \langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}}, \prec_{\mathcal{R}} \rangle$ where $\Delta^{\mathcal{R}} = \bigcup_{i=1}^{n} \Delta^{\mathcal{R}_i}$, $\cdot^{\mathcal{R}}$ is defined s.t. $A^{\mathcal{R}} = \bigcup_{i=1}^{n} A^{\mathcal{R}_i}$ and $R^{\mathcal{R}} = \bigcup_{i=1}^{n} R^{\mathcal{R}_i}$ for any concept name A (resp. role name R), and $\prec_{\mathcal{R}}$ is defined s.t. for each $x \in \Delta^{\mathcal{R}} \cap \Delta^{\mathcal{R}_1}$, $rk_{\mathcal{R}}(x) = rk_{\mathcal{R}_1}(x)$ and for each $x' \in \Delta^{\mathcal{R}} \cap \Delta^{\mathcal{R}_i}$, $rk_{\mathcal{R}}(x') = (max(\mathcal{R}_1) + ... + max(\mathcal{R}_{i-1})) + rk_{\mathcal{R}_i}(x')$ for $2 \leq i \leq n$.*

Definition 28 on Page 133 is analogous to Definition 27 on Page 132 except for the treatment of the ranks of elements of the interpretation domain. Taking the simple case of combining just two ranked interpretations, we keep the ranks, say 0 to k, of the first ranked interpretation $\mathcal{R}_1$ the same. Then, when we add the second ranked interpretation $\mathcal{R}_2$, we set the rank of the elements in the first "level" of $\mathcal{R}_2$ to $k + 1$ and those in the second rank to $k + 2$ etc. $(max(\mathcal{R}_1) + rk_{\mathcal{R}_2}(x))$. We can also prove that the disjoint union model

property extends to the vertical disjoint union of ranked models.

Lemma 3 (Vertical DU of Ranked Models are Ranked Models) *Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, a finite* ordered *set of* ranked models $\mathscr{R}$ *for $\langle \mathcal{T}, \mathcal{D} \rangle$ (all of whose domains are pairwise disjoint) and $\mathcal{R} = \boxminus \mathscr{R}$, then $\mathcal{R} \Vdash \langle \mathcal{T}, \mathcal{D} \rangle$.*

Proof: We prove the simplest case of combining two ranked models and then, by induction, the result extends to the general case. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two ranked models for $\langle \mathcal{T}, \mathcal{D} \rangle$ and let $\mathcal{R} = \mathcal{R}_1 \boxminus \mathcal{R}_2$ be their vertical disjoint union. Suppose that $\mathcal{R} \nVdash \langle \mathcal{T}, \mathcal{D} \rangle$. This means that for some $C \sqsubseteq D \in \mathcal{D}$, $\mathcal{R} \nVdash C \sqsubseteq D$. This, in turn, means that there is an $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \cap (\neg D)^{\mathcal{R}}$. Picking such an x we can show that there are only two cases. Either $x \in (C \sqcap \neg D)^{\mathcal{R}_1}$ or $x \in (C \sqcap \neg D)^{\mathcal{R}_2}$ because x, by Definition 28 on Page 133, has exactly one ranked interpretation from which it can originate.
Case 1: $x \in (C \sqcap \neg D)^{\mathcal{R}_1}$. Assume that $x \notin \min_{\prec_{\mathcal{R}_1}}(C^{\mathcal{R}_1})$, this means that there is a $y \in C^{\mathcal{R}_1}$ s.t. $rk_{\mathcal{R}_1}(y) < rk_{\mathcal{R}_1}(x)$. We pick a minimal such y, i.e., s.t. there is no $z \in C^{\mathcal{R}_1}$ s.t. $rk_{\mathcal{R}_1}(z) < rk_{\mathcal{R}_1}(y)$. Therefore, $y \in \min_{\prec_{\mathcal{R}_1}}(C^{\mathcal{R}_1})$. But $\Delta^{\mathcal{R}_1} \subseteq \Delta^{\mathcal{R}}$ and thus we know that $x, y \in \Delta^{\mathcal{R}}$. In addition, $C^{\mathcal{R}} = C^{\mathcal{R}_1} \cup C^{\mathcal{R}_2}$ because of Definition 28 on Page 133 and therefore $x, y \in C^{\mathcal{R}}$. Because the interpretation $\mathcal{R}_2$ is placed "on top of" $\mathcal{R}_1$ it means that the elements of $\mathcal{R}_1$ are lower than the elements of $\mathcal{R}_2$ in the combined interpretation $\mathcal{R}$. And because $y \in \min_{\prec_{\mathcal{R}_1}}(C^{\mathcal{R}_1})$, it must be the case that $y \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}})$ (there is no element of $\mathcal{R}$ that is more minimal than y, satisfying the same property). But this means that x is not a minimal element of $C^{\mathcal{R}}$ ($x \notin \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}})$) because y is more minimal than x. This is a contradiction with our earlier observation that $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \cap (\neg D)^{\mathcal{R}}$.
Case 2: $x \in (C \sqcap \neg D)^{\mathcal{R}_2}$. Assume that $x \notin \min_{\prec_{\mathcal{R}_2}}(C^{\mathcal{R}_2})$, this means that there is a $y \in C^{\mathcal{R}_2}$ s.t. $rk_{\mathcal{R}_2}(y) < rk_{\mathcal{R}_2}(x)$. We pick a minimal such y, i.e., s.t. there is no $z \in C^{\mathcal{R}_2}$ s.t. $rk_{\mathcal{R}_2}(z) < rk_{\mathcal{R}_2}(y)$. Therefore, $y \in \min_{\prec_{\mathcal{R}_2}}(C^{\mathcal{R}_2})$. It is important to note that there can be no C's in $\mathcal{R}_1$ because otherwise it would not be possible that the C's in the "top" interpretation $\mathcal{R}_2$ can be

minimal w.r.t. $\mathcal{R}$. Therefore, y must be a minimal element C w.r.t. $\mathcal{R}$. Just like Case 1, we have that x is not a minimal element of $C^{\mathcal{R}}$ ($x \notin \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}})$) anymore, because y is more minimal than x. Therefore, we have a contradiction with the assumption that $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \cap (\neg D)^{\mathcal{R}}$.

Therefore our supposition that $\mathcal{R} \not\Vdash \langle \mathcal{T}, \mathcal{D} \rangle$ is false and we accept that $\mathcal{R} \Vdash \langle \mathcal{T}, \mathcal{D} \rangle$. □

4.1.2 Exceptionality in Terms of Unsatisfiability

Recall that a concept C is exceptional w.r.t. a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ if $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \mathrel{\dot{\sqsubseteq}} \neg C$. It turns out that if our language is propositional (even restricting the $\mathcal{ALC}$ concept language defined in Section 2.1.1 to boolean operators and connectives), then checking exceptionality of C w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ corresponds to checking *(un)satisfiability* of C (in a classical sense) w.r.t. the classical counterpart of $\langle \mathcal{T}, \mathcal{D} \rangle$ [117, Lemma 5.21 and Corollary 5.22]. Here we take *propositional* $\mathcal{ALC}$ to define the following concept language:

$$C ::= A \mid \top \mid \bot \mid \neg C \mid C \sqcap C \mid C \sqcup C$$

As we can see, this language is a subset of the full concept language for $\mathcal{ALC}$ given in Section 2.1.1. We take the semantics of propositional $\mathcal{ALC}$ to be analogously defined to standard $\mathcal{ALC}$ (but restricted to the above vocabulary). Returning to the relationship between concept unsatisfiability and concept exceptionality, we restate the lemma showing their correspondence:

Lemma 4 (Exceptionality vs. Unsatisfiability in Propositional $\mathcal{ALC}$) *$\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \mathrel{\dot{\sqsubseteq}} \neg C$ if and only if $\mathcal{T} \cup \mathcal{D}' \models \top \sqsubseteq \neg C$ for any finite $\langle \mathcal{T}, \mathcal{D} \rangle$ and C represented in propositional $\mathcal{ALC}$.*

It is fairly straightforward to see that the introduction of roles in full $\mathcal{ALC}$ causes this correspondence to be broken:

Proposition 1 (Exceptionality vs. Unsatisfiability in $\mathcal{ALC}$) *$\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \mathrel{\sqsubseteq\!\!\!\sim} \neg C$ if and only if $\mathcal{T} \cup \mathcal{D}' \models \top \sqsubseteq \neg C$ for any finite $\langle \mathcal{T}, \mathcal{D} \rangle$ and C represented in $\mathcal{ALC}$.*

"$\Longrightarrow$": The contrapositive is: If $\mathcal{T} \cup \mathcal{D}' \not\models C \sqsubseteq \bot$ then $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r \top \mathrel{\sqsubseteq\!\!\!\sim} \neg C$. There is a model $\mathcal{I}$ for $\mathcal{T} \cup \mathcal{D}'$ s.t. there is an $x \in C^{\mathcal{I}}$. We can see that $\mathcal{I}$ can be considered a ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ where $\Delta^{\mathcal{R}} = \Delta^{\mathcal{I}}$, $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$ and $\prec_{\mathcal{R}} := rk : \Delta^{\mathcal{R}} \to \{0\}$. We have to show that (i) $\mathcal{R}$ is a ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$ and (ii) that $x^{\mathcal{R}}$ is such that $x^{\mathcal{R}} \in \min_{\prec_{\mathcal{R}}}(\Delta^{\mathcal{R}})$ and $x^{\mathcal{R}} \in C^{\mathcal{R}}$. For (i), we can see that $\mathcal{R}$ satisfies each $X \mathrel{\sqsubseteq\!\!\!\sim} Y \in \mathcal{D}$ because $\mathcal{I}$ satisfies each $X \sqsubseteq Y \in \mathcal{D}'$, $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$ and $\min_{\prec_{\mathcal{R}}}(X^{\mathcal{R}}) \subseteq X^{\mathcal{I}}$. For (ii), it is clear that $x^{\mathcal{R}} \in C^{\mathcal{R}}$ because $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$ and $x \in C^{\mathcal{I}}$. It is also clear that $x^{\mathcal{R}} \in \min_{\prec_{\mathcal{R}}}(\Delta^{\mathcal{R}})$ because $rk(x^{\mathcal{R}}) = 0$.
"$\Longleftarrow$": (counter-example) let $\mathcal{T} = \emptyset$, $\mathcal{D} = \{\top \mathrel{\sqsubseteq\!\!\!\sim} \neg X, C \mathrel{\sqsubseteq\!\!\!\sim} \exists R.X\}$ then, $\mathcal{T} \cup \mathcal{D}' \models \top \sqsubseteq \neg C$ but $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r \top \mathrel{\sqsubseteq\!\!\!\sim} \neg C$ because, considering the following ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$, where $\Delta^{\mathcal{R}} = \{a, b\}$, $C^{\mathcal{R}} = \{a\}$, $X^{\mathcal{R}} = \{b\}$, $R^{\mathcal{R}} = \{(a, b)\}$, $rk(a) = 0$ and $rk(b) = 1$, $\mathcal{R} \not\Vdash \top \mathrel{\sqsubseteq\!\!\!\sim} \neg C$. □

The fact that exceptionality doesn't correspond with unsatisfiability in full $\mathcal{ALC}$ is actually a positive consequence of the semantics of defeasible subsumption and exceptionality. In the counter-example to Proposition 1 on Page 137, the unsatisfiability of C is "caused" by the unsatisfiability of X (w.r.t. $\mathcal{T} \cup \mathcal{D}'$). This is justifiably intuitive because: "if all C's have at least one relationship to some X but there are no X's then there cannot be any C's either".

W.r.t. exceptionality, however, the exceptionality of X is not propagated to C (w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$). This is also justifiably intuitive because: "if X's are exceptional and C's have at least one relationship to some X then it doesn't mean that C is exceptional". That is, just because I am *related* to someone who is exceptional doesn't necessarily mean that I am exceptional as well.

While this behaviour bodes well for an intuitive semantics, the issue with roles that we highlighted above means that we still have the problem of re-

ducing exceptionality to classical DL entailment for full $\mathcal{ALC}$ (and any DL with roles). However, we do have some useful information from Proposition 1 on Page 137. That is, we know that the "left to right" direction of the correspondence does hold. This tells us that exceptionality is logically stronger than (classical) unsatisfiability. In other words, we know that exceptionality characterises a certain *kind* of unsatisfiability.

The counter-model pattern $\mathcal{R}$ for Proposition 1 on Page 137, illustrated in Figure 4.2, seems to suggest that unsatisfiabilities caused *directly* by a role filler being unsatisfiable are not included in this kind of unsatisfiability.

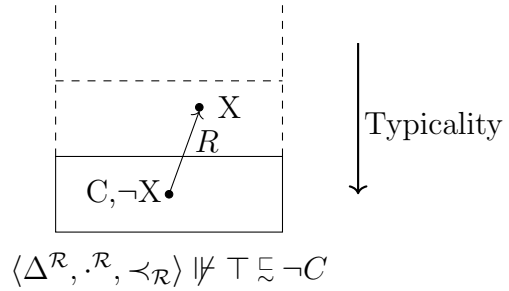


Figure 4.2: Intuitive semantics: exceptionality is not propagated through roles for the counter-example in Proposition 1.

Thus our general claim is: exceptionality corresponds to "propositional unsatisfiability" in $\mathcal{ALC}$. The remainder of this section is concerned with formalising "propositional unsatisfiability" for $\mathcal{ALC}$ and proving this claim.

Intuitively speaking, what we mean by propositional unsatisfiability is that we exclude cases where the unsatisfiability of a concept is derived directly from the unsatisfiability of a role filler in the KB (like the counter-example illustrated in Figure 4.2).

There is evidence in support of our claim because we notice that unsatisfiability and exceptionality correspond in the propositional case (Lemma 4 on Page 136), and therefore, it must be the case that the issue with roles is the *only* barrier to a correspondence for $\mathcal{ALC}$ because all the non-propositional features of $\mathcal{ALC}$ concern roles.

In order to exclude from consideration all those unsatisfiabilities caused by roles, we need a way of "localising" our view to individual objects in an interpretation domain without placing constraints from the KB on its *neighbourhood* (all other objects in the interpretation domain which are related to the given object via some role). We use *materialisation* and *internalisation* (well known in the context of propositional logic) to achieve this:

Definition 29 (Materialisation) *The* materialisation *of a classical DL subsumption $C \sqsubseteq D$ (resp. defeasible DL subsumption $C \mathrel{\tilde{\sqsubseteq}} D$) is $\neg C \sqcup D$. The* materialisation *of a set of classical DL subsumptions $\mathcal{T}$ (resp. defeasible DL subsumptions $\mathcal{D}$) is the set consisting of the materialisations for each subsumption in $\mathcal{T}$ (resp. $\mathcal{D}$). We denote the materialisation of a subsumption α by $mat(\alpha)$, and the materialisation of a set of subsumptions $\mathcal{K}$ by $mat(\mathcal{K})$.*

Definition 29 on Page 139 is central to defining the *internalisation* of a KB:

Definition 30 (Internalisation) *Let $\mathcal{K}$ be a set of DL subsumptions (each of which is either classical or defeasible), the* internalisation *of $\mathcal{K}$ is defined as: $\bigsqcap mat(\mathcal{K})$. We denote the* internalisation *of $\mathcal{K}$ by the concept $\mathcal{C}_{\mathcal{K}}$.*

The notion of internalisation allows us to isolate our view to objects in an interpretation without placing constraints on its neighbourhood. For example, given the KB $\langle \mathcal{T}, \mathcal{D} \rangle$ in the counter-example to Proposition 1 on Page 137, we have that C is *not* exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ but yet C is unsatisfiable w.r.t. $\mathcal{T} \cup \mathcal{D}$. This unsatisfiability is caused directly by the role filler for R being unsatisfiable. We would like to exclude these kinds of cases from consideration when characterising exceptionality for $\mathcal{ALC}$.

The good news is that when we internalise our knowledge to obtain $\mathcal{C}_{\mathcal{T} \cup \mathcal{D}'}$, we indeed exclude these cases because we find that $\not\models \mathcal{C}_{\mathcal{T} \cup \mathcal{D}'} \sqsubseteq \neg C$[1]. The

[1]Note that, in the propositional case, a classical KB and its internalised form correspond exactly in terms of their entailments. Indeed, this is the deduction theorem for propositional logics [91, 69, 189]. Our results are based on the fact that the deduction theorem does not generalise to $\mathcal{ALC}$ as illustrated by our examples.

statement $\models \mathcal{C}_{\mathcal{T}\cup\mathcal{D}'} \sqsubseteq \neg C$ means that the axiom $\mathcal{C}_{\mathcal{T}\cup\mathcal{D}'} \sqsubseteq \neg C$ is logically *valid* (or satisfied) in *any* interpretation for $\mathcal{C}_{\mathcal{T}\cup\mathcal{D}'}$ and C and, therefore, because we have that $\not\models \mathcal{C}_{\mathcal{T}\cup\mathcal{D}'} \sqsubseteq \neg C$, we know that there is an object z s.t. $z \in (\mathcal{C}_{\mathcal{T}\cup\mathcal{D}'})^{\mathcal{I}} \cap C^{\mathcal{I}}$ for some classical interpretation $\mathcal{I}$.

We give a simple example $\mathcal{I}$ for which this is the case: $\Delta^{\mathcal{I}} = \{a, b\}$, $X^{\mathcal{I}} = \{b\}$, $C^{\mathcal{I}} = \{a\}$ and $R^{\mathcal{I}} = \{(a, b)\}$. It is clear that $(\mathcal{C}_{\mathcal{T}\cup\mathcal{D}'})^{\mathcal{I}} = \{a\}$.

It is clear that internalisation, and the accompanying validity approach, does not place constraints on the neighbourhood of the object being considered. For example, in our interpretation $\mathcal{I}$ above, we see that the object b is in the neighbourhood of a and belongs to the concept $X^{\mathcal{I}}$. However, it is not possible for such a b to exist in a model for $\mathcal{T} \cup \mathcal{D}'$ since $\top \sqsubseteq \neg X$ appears in $\mathcal{T} \cup \mathcal{D}'$ and this constraint enforces that each object in each model for $\mathcal{T} \cup \mathcal{D}'$ cannot belong to the interpretation of X.

In particular from this approach, we notice that neighbourhood objects are allowed to violate *any* axiom in our defeasible KB (either in $\mathcal{T}$ or $\mathcal{D}'$). However, this behaviour does not accurately reflect the ranked model semantics of exceptionality. That is, it can be shown that given any $\langle \mathcal{T}, \mathcal{D} \rangle$, each ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ is s.t. for each $z \in \Delta^{\mathcal{R}}$, $z \in (\mathcal{C}_{\mathcal{T}})^{\mathcal{R}}$. The following lemma demonstrates this.

Lemma 5 (Ranked Models Classically Satisfy $\mathcal{T}$) *Let $\mathcal{R}$ be a ranked model for some defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$. Let $\mathcal{I}$ be the classical DL interpretation $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ such that $\Delta^{\mathcal{I}} = \Delta^{\mathcal{R}}$ and $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$. Then $\mathcal{I}$ is a model for $\mathcal{T}$.*

Proof: We know that $\mathcal{R} \Vdash \mathcal{T}$ because $\mathcal{R} \Vdash \langle \mathcal{T}, \mathcal{D} \rangle$. Assume that $\mathcal{I}$ is not a model for $\mathcal{T}$. Therefore, there is a $C \sqsubseteq D \in \mathcal{T}$ s.t. there is an $x \in C^{\mathcal{I}}$ and $x \notin D^{\mathcal{I}}$. This must mean that $x \in C^{\mathcal{R}}$ and $x \notin D^{\mathcal{R}}$ because $\Delta^{\mathcal{I}} = \Delta^{\mathcal{R}}$ and $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$. But this is a contradiction with $\mathcal{R} \Vdash \mathcal{T}$. Therefore, $\mathcal{I} \Vdash \mathcal{T}$. □

Lemma 5 on Page 140 shows that every element in a ranked model for a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ must satisfy each axiom in $\mathcal{T}$. Therefore, applied to our context of internalising knowledge from the defeasible KB, it should only

be the *defeasible* subsumptions in the KB that can be violated by exceptional objects (whereas our classical subsumptions should not be violated).

The following example illustrates this: let $\mathcal{T} = \{A \sqsubseteq \exists R.B, \exists R.X \sqsubseteq \neg C, B \sqsubseteq X\}$ and $\mathcal{D} = \{B \mathrel{\tilde{\sqsubseteq}} \neg C, \top \mathrel{\tilde{\sqsubseteq}} A \sqcup B\}$. We can see that C is exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ but that $\not\models \mathcal{C}_{\mathcal{T} \cup \mathcal{D}'} \sqsubseteq \neg C$. The following interpretation $\mathcal{I}$ demonstrates that the latter validity does not hold: $\Delta^{\mathcal{I}} = \{a, b\}$, $A^{\mathcal{I}} = C^{\mathcal{I}} = \{a\}$, $B^{\mathcal{I}} = \{b\}$, $X^{\mathcal{I}} = \emptyset$ and $R^{\mathcal{I}} = \{(a, b)\}$. It is clear that b is allowed to violate the axiom $B \sqsubseteq X$ and this is what causes us to lose the validity $\models \mathcal{C}_{\mathcal{T} \cup \mathcal{D}'} \sqsubseteq \neg C$.

Thus, we investigate the approach of internalising purely the *defeasible* subsumptions contained in $\mathcal{D}$. We arrive at a formal definition for propositional unsatisfiability for preferential $\mathcal{ALC}$ (and preferential DLs in general):

Definition 31 (Propositional Unsatisfiability for Preferential DLs) *Let $\mathcal{T} \cup \mathcal{D}'$ be the classical counterpart of a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, and C a concept, C is* propositionally unsatisfiable *w.r.t. $\mathcal{T} \cup \mathcal{D}'$ if $\mathcal{T} \models \mathcal{C}_{\mathcal{D}'} \sqsubseteq \neg C$.*

Given Definition 31 on Page 141, we now formalise our claim about the correspondence between exceptionality and propositional unsatisfiability. Even though this claim turns out to be false, it is not as a result of some deficiency in Definition 31. Rather, it is that propositional unsatisfiability is slightly too weak to capture exceptionality in full $\mathcal{ALC}$. It actually turns out that there is a special kind of exceptionality (captured in Definition 15 on Page 92) called total exceptionality which *can* be propagated through roles.

Proposition 2 *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a defeasible KB and C a concept. Then, C is exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ if and only if C is propositionally unsatisfiable w.r.t. $\mathcal{T} \cup \mathcal{D}'$.*

" $\Longrightarrow$ " (counter-example) $\mathcal{T} = \{X \sqsubseteq Y\}$, $\mathcal{D} = \{X \mathrel{\tilde{\sqsubseteq}} \neg Y, C \mathrel{\tilde{\sqsubseteq}} \exists R.X\}$. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \mathrel{\tilde{\sqsubseteq}} \neg C$ but $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}'} \sqsubseteq \neg C$. □

The counter-example for Proposition 2 on Page 141 illustrates the occurrence of totally exceptional LHS concepts of defeasible subsumptions. That is, in the example, the concept X is totally exceptional w.r.t. the KB.

We have argued in Section 2.8.2 that these kinds of concepts are usually the result of modelling errors in ontology development because it represents a form of logical incoherence (see Definition 15 on Page 92). This is essentially the representation of defeasible information which actually "behaves" as strict information to cause logical incoherence.

For example, the axioms $C \mathrel{\tilde{\sqsubseteq}} \bot$ and $C \sqsubseteq \bot$ have the same ranked models. Therefore, representing $C \mathrel{\tilde{\sqsubseteq}} \bot$ in the DTBox is actually the same as representing $C \sqsubseteq \bot$ in the TBox. Hence, in the above example, the axiom $X \mathrel{\tilde{\sqsubseteq}} \neg Y$ is masquerading as defeasible information whereas it should be treated as strict information since $\langle \mathcal{T}, \mathcal{D} \rangle = \langle \{X \sqsubseteq Y\}, \{X \mathrel{\tilde{\sqsubseteq}} \neg Y, C \mathrel{\tilde{\sqsubseteq}} \exists R.X\} \rangle$ is *ranked equivalent* to $\langle \hat{\mathcal{T}}, \hat{\mathcal{D}} \rangle = \langle \{X \sqsubseteq Y, X \sqsubseteq \neg Y\}, \{C \mathrel{\tilde{\sqsubseteq}} \exists R.X\} \rangle$ (both KBs have the same ranked models).

Unfortunately, it appears that there is no (declarative) reduction to classical DL entailment for total exceptionality. However, we will show in Section 4.2 that there is a useful operational reduction. We now show that eliminating the case of total exceptionality (and only this case), gives us a clean and declarative reduction to classical DL entailment for exceptionality.

We restrict ourselves to $\langle \mathcal{T}, \mathcal{D} \rangle$'s in which each LHS concept of the subsumptions in $\mathcal{D}$ are *not* totally exceptional. We define such $\langle \mathcal{T}, \mathcal{D} \rangle$'s as *left hand side coherent* (*LHS-coherent*):

Definition 32 (LHS-coherent) *A defeasible KB, $\langle \mathcal{T}, \mathcal{D} \rangle$, is* LHS-coherent *if C is not totally exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ for each $C \mathrel{\tilde{\sqsubseteq}} D \in \mathcal{D}$. $\langle \mathcal{T}, \mathcal{D} \rangle$ is* LHS-incoherent *if it is not LHS-coherent.*

This leads to a positive result and reduction to classical DL entailment for exceptionality:

Theorem 1 (Reducing Exceptionality to Classical DL Entailment) *Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ which is LHS-coherent, and a concept C, C is exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ if and only if C is propositionally unsatisfiable w.r.t. $\mathcal{T} \cup \mathcal{D}'$.*

Proof: We replace the above statement by its contraposition:

$\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}'} \sqsubseteq \neg C$ if and only if $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r \top \sqsubseteq\!\!\!\sim \neg C$.

" $\Longrightarrow$ " We pick a model $\mathcal{I}$ of $\mathcal{T}$ s.t. there exists an $x \in (\mathcal{C}_{\mathcal{D}'})^{\mathcal{I}} \cap C^{\mathcal{I}}$. We will show that we can construct a ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ from the information in $\mathcal{I}$ s.t. there is a $y \in \min_{\prec_{\mathcal{R}}}(\Delta^{\mathcal{R}}) \cap C^{\mathcal{R}}$ corresponding to the element x in $\Delta^{\mathcal{I}}$. Our strategy is as follows: we know that $\langle \mathcal{T}, \mathcal{D} \rangle$ is LHS-coherent so we know that for each $X \sqsubseteq\!\!\!\sim Y \in \mathcal{D}$ there is a ranked model $\mathcal{R}_X$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. $X^{\mathcal{R}}$ is non-empty. We pick such a ranked model $\mathcal{R}_X$ for each $X \sqsubseteq\!\!\!\sim Y \in \mathcal{D}$ and let $\mathscr{R}$ be the set containing these models and only these models (i.e., $\bigcup\{\mathcal{R}_X\}$ for each $X \sqsubseteq\!\!\!\sim Y \in \mathcal{D}$). We obtain the horizontal disjoint union of these models $\mathcal{R}_{\langle \mathcal{T}, \mathcal{D} \rangle} = \boxplus\mathscr{R}$ (Definition 27 on Page 132). Lemma 2 on Page 133 tells us that $\mathcal{R}_{\langle \mathcal{T}, \mathcal{D} \rangle}$ is a model for $\langle \mathcal{T}, \mathcal{D} \rangle$. But we also know that $\mathcal{I}$ can be considered as a ranked interpretation $\mathcal{R}_{\mathcal{I}}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ where $\Delta^{\mathcal{R}_{\mathcal{I}}} = \Delta^{\mathcal{I}}$, $\cdot^{\mathcal{R}_{\mathcal{I}}} = \cdot^{\mathcal{I}}$, $rk_{\mathcal{R}_{\mathcal{I}}}(z) = 1$ for each $z \in \Delta^{\mathcal{R}_{\mathcal{I}}} \backslash \{x\}$ and $rk_{\mathcal{R}_{\mathcal{I}}}(x) = 0$. Let $\mathcal{R} = \mathcal{R}_{\langle \mathcal{T}, \mathcal{D} \rangle} \boxplus \mathcal{R}_{\mathcal{I}}$ (we place $\mathcal{R}_{\mathcal{I}}$ on top of $\mathcal{R}_{\langle \mathcal{T}, \mathcal{D} \rangle}$). We make one slight modification to the ranking function of $\mathcal{R}$. That is, we set $rk_{\mathcal{R}}(x) = 0$ (we push x to the bottom level of $\mathcal{R}$). It is clear that every element of $\Delta^{\mathcal{R}}$ satisfies $\mathcal{T}$ because we have taken the disjoint unions of interpretations and we have not changed the interpretation function for $\mathcal{R}$. It is clear that each element of $\Delta^{\mathcal{R}}$ satisfies $\mathcal{D}$ because the most minimal X objects (for each $X \sqsubseteq\!\!\!\sim Y \in \mathcal{D}$) in $\mathcal{R}$ satisfy $\mathcal{D}$ because $\mathcal{R}_{\langle \mathcal{T}, \mathcal{D} \rangle}$ is a model for $\langle \mathcal{T}, \mathcal{D} \rangle$. We also note that $x \in \mathcal{C}^{\mathcal{R}}_{\mathcal{D}'}$ and so x also satisfies $\mathcal{D}$.

" $\Longleftarrow$ " There exists a ranked model, $\mathcal{R}$, such that $\mathcal{R} \Vdash \langle \mathcal{T}, \mathcal{D} \rangle$ and there exists an $x \in C^{\mathcal{R}}$ s.t. $rk_{\mathcal{R}}(x) = 0$. We have to show that there exists a model $\mathcal{I}$ for $\mathcal{T}$ such that there exists a $y \in {\mathcal{C}_{\mathcal{D}'}}^{\mathcal{I}} \cap C^{\mathcal{I}}$. We define an interpretation

$\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ s.t. $\Delta^{\mathcal{I}} = \Delta^{\mathcal{R}}$ and $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$. From Lemma 5 on Page 140 we know that $\mathcal{I} \Vdash \mathcal{T}$. We select the element $y \in \Delta^{\mathcal{I}}$ s.t. $y = x$. All we need to prove is that $y \in {\mathcal{C}_{\mathcal{D}'}}^{\mathcal{I}}$. Let us assume that $y \notin {\mathcal{C}_{\mathcal{D}'}}^{\mathcal{I}}$. This means that there exists a $X \sqsubseteq Y \in \mathcal{D}$ such that $y \in X^{\mathcal{I}}$ and $y \notin Y^{\mathcal{I}}$. This must mean that the corresponding x in $\Delta^{\mathcal{R}}$ is an exceptional element of $X^{\mathcal{R}}$ and therefore it cannot reside on the bottom-most level of $\mathcal{R}$. That is, this is a contradiction with the fact that $rk_{\mathcal{R}}(x) = 0$. Therefore $y \in {\mathcal{C}_{\mathcal{D}'}}^{\mathcal{I}}$. □

We now have an understanding of how exceptionality can be reduced to classical DL reasoning. In the next section we will show an interesting and useful a priori ordering of axioms in a defeasible KB based on exceptionality. Furthermore, we present an algorithm for computing this a priori ordering for a given defeasible KB. As we shall see, this *precompiled view* of a defeasible KB is required for all the defeasible entailment algorithms that we present in this Chapter (Section 4.3 onwards).

4.2 Ranking of a Defeasible Knowledge Base

As we saw in Chapter 2 there are various approaches for permitting exceptions in DL reasoning. It becomes apparent in the presentation of these approaches that there are some basic issues, with regards to exceptions, that *all* such approaches should address. One of these issues is that there must be some way of handling priorities among defeasible statements in the formalism under consideration.

Most of the presented approaches in Chapter 2 (except for Rational Closure and similar entailment regimes) choose to fully "externalise" the priority relation among defeasible statements. That is, they choose to fully hand over the specification of this relation to the ontology engineer. While this may give desirable flexibility to the user to decide which information should override other information, we argue that this relation should not be completely arbitrary. Indeed, many of the proponents of these formalisms agree on *specificity*

as one property that should always be satisfied by this relation.

For example, if students usually don't pay taxes and employed students usually *do* pay taxes then, assuming we want to draw inferences about the tax paying status of students in general, we can apply the first "default" to conclude that students usually don't pay taxes. This is certainly the most reasonable inference to make if we have no other information. On the other hand, suppose we want to draw generalised inference about the tax paying status of *employed* students, then *both* above defaults become applicable (employed students are also students). However, applying both defaults leads to the contradictory situation that employed students pay taxes and *don't* pay taxes.

Clearly one would rather prefer applying the default that takes into account the most specific (and therefore the most complete) knowledge relevant to the query. That is, if we are asking about general students it should be safe to conclude that they generally don't pay taxes (applying the first default). Whereas, if we are asking about employed students, knowledge about employed students *specifically* should take precedence over knowledge about more general students (if there is a clash with applying both defaults).

Bonatti et al. [27] characterise priority relations on defaults that are *defined* by *specificity*: let $\mathcal{K} = \mathcal{S} \cup \mathcal{D}$ be a finite knowledge base of DIs ($\mathcal{D}$) and SIs ($\mathcal{S}$). Then the priority relation among the DIs in $\mathcal{D}$ (which we denote by $\prec$) is *defined* by *specificity* if: for any $\delta_1 = C_1 \sqsubseteq_n D_1$ and $\delta_2 = C_2 \sqsubseteq_n D_2$ s.t. $\delta_1, \delta_2 \in \mathcal{D}$, $\delta_1 \prec \delta_2$ (it is preferred to apply δ_1 over δ_2) if and only if $\mathcal{S} \models C_1 \sqsubseteq C_2$ and $\mathcal{S} \not\models C_2 \sqsubseteq C_1$.

The above version of specificity intuitively says that δ_1 has a higher priority than δ_2 if and only if C_1 is more specific than C_2 (according to the strict information in the KB). We note that it is possible to define several variants of this property depending on which approach to defeasible reasoning we are dealing with. For example, we may not want the priority relation to be *defined* by specificity (this might be too strong). However, we may just want our priority relation to *respect* specificity. In other words, the "right-to-left"

direction of the above definition by Bonatti et al. [27] may be enforced (but not the other direction). We may also choose to involve the defeasible information (in addition to the strict information) when determining specificity. In fact, we define such a variant at the end of this section for the preferential reasoning context.

It can be convincingly argued that, at the very least, some version of specificity should be satisfied by the priority relation on defaults. Interestingly, preferential reasoning mechanisms such as Rational Closure (see Section 2.8) abide by this philosophy. We will show in this section, and more broadly in this chapter, that such approaches *presume* that specificity should be respected. That is, the property of specificity is "internalised" in the semantics of Rational Closure and similar constructions.

In fact, the first step in all the procedures we present in this chapter (including Rational Closure) is to *compute* the ordering on the defeasible subsumptions in the defeasible KB. It is possible to compute the ordering because it is induced by exceptionality (see Definition 14 on Page 91). Consider the following example:

Example 15 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$*:*

$$\left\langle \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\}, \left\{ \begin{array}{lcl} \mathsf{Student} & \mathrel{\tilde{\sqsubseteq}} & \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice}), \\ \mathsf{EmployedStudent} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{receives}.\mathsf{TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \mathrel{\tilde{\sqsubseteq}} & \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice}) \end{array} \right\} \right\rangle$$

□

Intuitively speaking, the information represented in Example 15 on Page 146 is that employed students are types of students, students usually aren't obliged to pay taxes, employed students usually *are* obliged, and employed students who are parents usually aren't obliged again.

This example illustrates that employed students are *exceptional* types of students (w.r.t. paying taxes), and employed students who are parents are exceptional types of *employed* students (again w.r.t. paying taxes). Informally then, we can build an "exceptionality chain" from the antecedent concepts in

the defeasible subsumptions of $\langle \mathcal{T}, \mathcal{D} \rangle$. That is, we can conceive of the chain $\mathsf{Student} \prec_{\langle \mathcal{T}, \mathcal{D} \rangle} \mathsf{EmployedStudent} \prec_{\langle \mathcal{T}, \mathcal{D} \rangle} \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$ representing the fact that students are the least exceptional entities in $\langle \mathcal{T}, \mathcal{D} \rangle$, employed students are exceptional students, and employed students that are parents are exceptional employed students.

Using the semantics of exceptionality and defeasible subsumption, we can confirm the correctness of this chain. It is easy to see that one can "realise" or instantiate a $\mathsf{Student}$ object on the bottom-most level of a ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$ ($\mathsf{Student}$ is not exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$). One cannot expect $\mathsf{EmployedStudent}$ objects to be able to reside on the bottom-most level as well because it would lead to a contradictory situation ($\mathsf{EmployedStudent}$ is exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$). Therefore, the lowest level that these objects can occur on is the second-level of a ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$. Similarly, the lowest possible level for $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$ objects is the third level of a ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$.

Example 15 on Page 146 demonstrates that we can compute an ordering of the antecedent concepts of the defeasible subsumptions in a defeasible KB. Recall that from Definition 14 on Page 91, exceptionality of an antecedent concept of a subsumption extends to the subsumption as a whole. Therefore, we can order sentences in our KB according to degree of exceptionality.

We observe that while $\mathsf{Student}$ is not exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ in Example 15 on Page 146, both $\mathsf{EmployedStudent}$ and $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$ are exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$. However, notice that if we ignore the constraint about general students in $\langle \mathcal{T}, \mathcal{D} \rangle$ ($\mathsf{Student} \sqsubseteq \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$), then $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$ still remains exceptional w.r.t. the remainder constraints. $\mathsf{EmployedStudent}$, however, is no longer exceptional.

Informally, this behaviour illustrates that $\mathsf{EmployedStudent}$ is exceptional w.r.t. the properties of a general student (removing constraints about general students allows one to consider employed students as prototypically normal and unexceptional). In contrast, $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$ remains exceptional w.r.t. the remaining constraints about employed students (the

former don't pay taxes while the latter do). However, we notice that employed students that are parents are consistent with the properties of general students (both of them don't pay taxes), and therefore removing constraints about general students does not impact on their exceptionality.

In summary, the "exception-to-exception" ordering of a defeasible KB, as described above, can be computed using an iterative procedure. The procedure includes a core sub-routine (called Exceptional) to capture all the exceptional subsumptions from a given set of defeasible subsumptions (see Definition 17 on Page 94). Note that in this section, we first concentrate on LHS-coherent KBs $\langle \mathcal{T}, \mathcal{D} \rangle$ (see Definition 32 on Page 142). Recall that LHS-coherent KBs are those where the LHS concepts of all defeasible subsumptions are s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r C \mathrel{\tilde{\sqsubseteq}} \bot$. Later in this section we will address general $\langle \mathcal{T}, \mathcal{D} \rangle$'s that are possibly LHS-*incoherent*.

Procedure `Exceptional`$(\mathcal{T}, \mathcal{E})$

Input: A set of strict subsumptions $\mathcal{T}$ and a set of defeasible subsumptions $\mathcal{E}$ s.t. $\mathcal{T} \cup \mathcal{E}$ is LHS-coherent.

Output: $\mathcal{E}_{exc} \subseteq \mathcal{E}$ such that $\mathcal{E}_{exc}$ is exceptional w.r.t. $\mathcal{T} \cup \mathcal{E}$

1 $\mathcal{E}_{exc} := \emptyset$;
2 **foreach** $C \mathrel{\tilde{\sqsubseteq}} D \in \mathcal{E}$ **do**
3 **if** $\mathcal{T} \models \mathcal{C}_{\mathcal{E}} \sqsubseteq \neg C$ **then**
4 $\mathcal{E}_{exc} := \mathcal{E}_{exc} \cup \{C \mathrel{\tilde{\sqsubseteq}} D\}$;
5 **return** $\mathcal{E}_{exc}$;

Note that $\mathcal{C}_{\mathcal{E}}$ in Line 3 represents the internalisation of the set $\mathcal{E}$ (see Definition 30 on Page 139) and the entailment check $\mathcal{T} \models \mathcal{C}_{\mathcal{E}} \sqsubseteq \neg C$ represents the reduction to classical DL entailment for checking exceptionality of C w.r.t. $\mathcal{T} \cup \mathcal{E}$ (see Theorem 1 on Page 143). Observing this entailment check we note that $\mathcal{T}$ (the strict subsumptions) can also contribute to the exceptionality of a concept and, therefore, we take this information into account when computing the exceptional subsumptions of a given set.

It is clear that Procedure `Exceptional` on Page 148 terminates because $\mathcal{E}$ is finite. Its soundness (all the elements of $\mathcal{E}_{exc}$ are exceptional w.r.t. $\mathcal{E}$) follows from the soundness of our reduction of exceptionality to classical DL entailment (see Theorem 1 on Page 143). We use $\langle \mathcal{T}, \mathcal{D} \rangle$ from Example 15 on Page 146 to illustrate the behaviour of Procedure `Exceptional`:

Example 16 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{E} \rangle$*:*

$$\left\langle \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\}, \left\{ \begin{array}{lcl} \mathsf{Student} & \lesssim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} & \lesssim & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \lesssim & \neg(\exists \mathsf{receives.TaxInvoice}) \end{array} \right\} \right\rangle$$

After executing Procedure `Exceptional` *on Page 148 on* $\langle \mathcal{T}, \mathcal{E} \rangle$ *we obtain:*

$$\mathcal{E}_{exc} = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \lesssim & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \lesssim & \neg(\exists \mathsf{receives.TaxInvoice}) \end{array} \right\}$$

□

In Example 16 on Page 149 we see that EmployedStudent and EmployedStudent $\sqcap$ $\exists$hasChild.$\top$ are the only known antecedent concepts that are exceptional w.r.t. $\langle \mathcal{T}, \mathcal{E} \rangle$. Therefore, we move the defeasible subsumptions in which they occur on the LHS to the exceptional set $\mathcal{E}_{exc}$.

Example 16 on Page 149 shows that one can recursively apply Procedure `Exceptional` on Page 148 to obtain the exceptionality sequence (see Definition 18 on Page 94) of a defeasible KB. From this exceptionality sequence we can determine the *ranking* of a defeasible KB.

Definition 33 (Ranking of a Defeasible KB) *Let* $\langle \mathcal{T}, \mathcal{D} \rangle$ *be a defeasible KB, then the* ranking *for* $\langle \mathcal{T}, \mathcal{D} \rangle$ *is a sequence of subsets* $\mathcal{D}_0, \ldots, \mathcal{D}_n$ *of* $\mathcal{D}$ *s.t. for each* $0 \leq i \leq n$*, for each* $C \lesssim D \in \mathcal{D}_i$*,* $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_p}(C) = i$ *(see Definition 19 on Page 95).*

Definition 33 on Page 149 characterises an ordering of the defeasible subsumptions in the KB according to the rank of their antecedent concepts which, in turn, is determined by their degree of exceptionality.

We will show now that there is only one minor difference between the ranking of a defeasible KB and its exceptionality sequence, from the perspective of their operational definitions. That is, when we move to the next subset in the sequence, the ranking "cuts" the exceptional axioms from the previous subset and "pastes" them into the next subset, whereas the exceptionality sequence "copies" these axioms into the next subset in the sequence.

The procedure that we now present for computing the ranking of a defeasible KB is based on Procedure `Exceptional` on Page 148. We first consider the case where the input defeasible KB is LHS-coherent (see Definition 32 on Page 142) and later address general defeasible KBs.

Procedure `ComputeRankingA`$(\langle \mathcal{T}, \mathcal{D} \rangle)$

Input: A *LHS-coherent* defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$.

Output: The ranking $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$.

1 $R{:=}\emptyset$;
2 $\mathcal{E}_0 := \mathcal{D}$;
3 $\mathcal{E}_1 :=$ `Exceptional`$(\mathcal{T}, \mathcal{E}_0)$;
4 $i := 0$;
5 **while** $\mathcal{E}_{i+1} \neq \emptyset$ **do**
6 $i := i + 1$;
7 $\mathcal{E}_{i+1} :=$ `Exceptional`$(\mathcal{T}, \mathcal{E}_i)$;
8 **for** $j = 1$ *to* i **do**
9 $\mathcal{D}_{j-1} := \mathcal{E}_{j-1} \setminus \mathcal{E}_j$;
10 $R := R \cup \{\mathcal{D}_{j-1}\}$;
11 **return** R;

Observe that Lines 1 to 7 of Procedure `ComputeRankingA` on Page 150 compute the exceptionality sequence for $\langle \mathcal{T}, \mathcal{D} \rangle$ and Lines 8 to 10 do the "cutting" and "pasting", as described above, to arrive at the ranking for $\langle \mathcal{T}, \mathcal{D} \rangle$. We briefly illustrate the behaviour of Procedure `ComputeRankingA` with an example.

Example 17 *Consider the original defeasible KB* $\langle \mathcal{T}, \mathcal{E} \rangle$ *in Example 16 on Page 149:*

$$\left\langle \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\}, \left\{ \begin{array}{lll} \mathsf{Student} & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} & \precsim & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}) \end{array} \right\} \right\rangle$$

Executing Lines 1 *to* 7 *of Procedure* `ComputeRankingA` *on Page 150, we obtain:*

$$\mathcal{E}_0 = \left\{ \begin{array}{lll} \mathsf{Student} & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} & \precsim & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}) \end{array} \right\},$$

$$\mathcal{E}_1 = \left\{ \begin{array}{lll} \mathsf{EmployedStudent} & \precsim & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}) \end{array} \right\},$$

$$\mathcal{E}_2 = \left\{ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \precsim \neg(\exists \mathsf{receives.TaxInvoice}) \right\},$$

$$\mathcal{E}_3 = \emptyset$$

Thereafter, executing Lines 8 *to* 10 *on this sequence we obtain the ranking:*

$$\begin{array}{lll} \mathcal{D}_0 = \{ & \mathsf{Student} \precsim \neg(\exists \mathsf{receives.TaxInvoice}) & \}, \\ \mathcal{D}_1 = \{ & \mathsf{EmployedStudent} \precsim \exists \mathsf{receives.TaxInvoice} & \}, \\ \mathcal{D}_2 = \{ & \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \precsim \neg(\exists \mathsf{receives.TaxInvoice}) & \} \end{array}$$

□

We have to show that Procedure `ComputeRankingA` on Page 150 actually terminates and that it is sound (i.e., the computed ranking corresponds with its semantic definition). Notice that to show termination we have to show that the condition on Line 5 is actually falsified at some point. That is, we should obtain an i s.t. $\mathcal{E}_{i+1} = \emptyset$. To show this, it is sufficient to demonstrate that $\mathcal{E}_{i+1}$ becomes smaller with every iteration of the while loop.

Formally, this amounts to demonstrating that: $\mathcal{E}_{i+1} \subset \mathcal{E}_i$ for each $0 \leq i \leq n-1$. Observe that it can never be the case that $\mathcal{E}_{i+1} \supset \mathcal{E}_i$ for any such i

because the procedures are only *retaining* the exceptional items from some set in the next set of the sequence (the procedure does not *add* any information to the set). Therefore, the only other possibility which contradicts with what we have to prove is $\mathcal{E}_{i+1} = \mathcal{E}_i$ for some i. Intuitively this case means that *all* axioms in the given set are exceptional w.r.t. the given defeasible KB. The following lemma helps to prove that this is impossible.

Lemma 6 ($\mathcal{E}_{i+1} \neq \mathcal{E}_i$ for LHS-coherent KBs) *Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, if $\langle \mathcal{T}, \mathcal{D} \rangle$ is LHS-coherent then there is a $C \sqsubseteq D \in \mathcal{D}$ s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r \top \sqsubseteq \neg C$.*

Proof: Suppose that $\langle \mathcal{T}, \mathcal{D} \rangle$ is LHS-coherent and, contrary to the lemma statement, that for each $C \sqsubseteq D \in \mathcal{D}$ s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq \neg C$. We proceed by trying to derive a contradiction from this. For each $C \sqsubseteq D \in \mathcal{D}$ we pick a ranked model $\mathcal{R}_C$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. there is an $x \in C^{\mathcal{R}_C}$ (we know that such ranked models exist because $\langle \mathcal{T}, \mathcal{D} \rangle$ is LHS-coherent). Now, we take the horizontal union of these ranked models $\mathcal{R} = \boxplus \mathcal{R}_C$ (see Definition 27 on Page 132). $\mathcal{R}$ is also a ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$ by Lemma 2 on Page 133. We pick a $C \sqsubseteq D \in \mathcal{D}$ s.t. $rk_{\mathcal{R}}(C) = min\{rk_{\mathcal{R}}(X) \mid X \sqsubseteq Y \in \mathcal{D}\}$ (there must be a minimal rank because of the modular ordering on the elements of ranked models). Now, we pick an element $y \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}})$. From our supposition we know that C is exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ and therefore, $rk_{\mathcal{R}}(y) > 0$. However, notice that the minimality of y's rank suggests that $y \notin (X \sqcap \neg Y)^{\mathcal{R}}$ for each $X \sqsubseteq Y \in \mathcal{D}$. That is, it is the most minimal element in $\mathcal{R}$ that belongs to any $X^{\mathcal{R}}$ s.t. $X \sqsubseteq Y \in \mathcal{D}$. We also know that there is no element z on the bottom-most rank of $\mathcal{R}$ s.t. $z \in X^{\mathcal{R}}$ for some $X \sqsubseteq Y \in \mathcal{D}$ (because all the axioms in $\mathcal{D}$ are exceptional). However, this implies that the rank of y is not minimal w.r.t. each ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$. In other words, we can "push" y to the bottom-most rank in $\mathcal{R}$ (let $rk_{\mathcal{R}}(y) = 0$). All other things equal, the modified $\mathcal{R}$ will still be a ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$ because it does not violate any axiom in $\mathcal{D}$ as described earlier. However, the resulting modified $\mathcal{R}$ is now s.t. $\mathcal{R} \not\Vdash \top \sqsubseteq \neg C$. Therefore, our supposition is false. That is, we

have found a ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ and a $C \sqsubseteq D \in \mathcal{D}$ s.t. $\mathcal{R} \not\Vdash \top \sqsubseteq \neg C$. $\square$

Notice that Lemma 6 on Page 152 proves that $\mathcal{E}_{i+1} \subset \mathcal{E}_i$ for $i = 0$. In other words, we have proven that because $\mathcal{E}_0$ is LHS-coherent (as stipulated in our procedure), it is guaranteed to contain non-exceptional axioms. Hence, because such axioms are removed from $\mathcal{E}_0$ when moving to $\mathcal{E}_1$, $\mathcal{E}_1$ is guaranteed to be smaller than $\mathcal{E}_0$. It is easy to see, by induction, that this holds for $0 \leq i \leq n-1$ because of *monotonicity* of classical reasoning (our procedure is reduced to classical entailment). That is, if $\mathcal{E}_i$ is LHS-coherent, then $\mathcal{E}_{i+1}$ is LHS-coherent (removing axioms cannot cause us to gain LHS-incoherence).

The soundness for Procedure `ComputeRankingA` on Page 150 has been shown in an independent effort to our own [71, Proposition 13]. Essentially, this result shows the correspondence between the semantic (Definition 12 on Page 87) and procedural (Definition 19 on Page 95) definitions for the rank of a concept. We rephrase their proof statement here to make it clear to the reader what was required to be shown using our own terminology.

Lemma 7 (Soundness of Procedure `ComputeRankingA`) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB and $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its computed ranking (according to Procedure `ComputeRankingA` on Page 150). Then, $C \sqsubseteq D \in \mathcal{D}_i$ if and only if $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C) = i$ for each $0 \leq i \leq n$.*

We now turn our attention to general defeasible KBs. That is, we permit the possibility that the KB is *not* LHS-coherent. Recall from Definitions 15 and 32 that a LHS-*incoherent* KB $\langle \mathcal{T}, \mathcal{D} \rangle$ is s.t. C is ranked unsatisfiable w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ for some $C \sqsubseteq D \in \mathcal{D}$.

It turns out that LHS-incoherent KBs cannot be handled purely by the standard ranking procedure (Procedure `ComputeRankingA` on Page 150). That is, a modification of this procedure is required. The reason for this relates to the semantics of defeasible subsumption and specifically to the semantics of ranked unsatisfiability of a concept.

We recall that $C \precsim \bot$ and $C \sqsubseteq \bot$ have the same ranked models (i.e., they are *ranked equivalent* axioms). It is easy to see, then, that one can encode the latter classical subsumption using purely defeasible subsumptions. For example, the set $\{C \precsim D, C \precsim \neg D\}$ encodes the same information as $C \precsim \bot$ and, therefore, $C \sqsubseteq \bot$ (which is also equivalent to $\{C \sqsubseteq D, C \sqsubseteq \neg D\}$).

While this behaviour is acceptable (and arguably intuitive) from a semantic perspective, it turns out to be problematic when reducing total exceptionality to classical DL entailment (see Proposition 2 on Page 141). It is actually not too difficult to notice why this is the case. We recall from Section 4.1 that our reduction of exceptionality to classical DL entailment treats the strict information $\mathcal{T}$ and the defeasible information D differently.

That is, $\mathcal{T}$ is applied as "global" constraints in the ranked models and $\mathcal{D}$ is applied "locally" to objects in the models (ignoring neighbourhood objects). It is clear then, that if we represent information such as $\{C \precsim D, C \precsim \neg D\}$ in our KB (i.e., strict information that is just masquerading as defeasible information), then we would not treat these constraints globally in our reduction as they should be. Of course, this may in turn lead to incorrect inferences. The phenomenon that $\{C \precsim D, C \precsim \neg D\}$ is actually equivalent to $\{C \sqsubseteq D, C \sqsubseteq \neg D\}$ is called *hidden strict information*. It has been shown in Section 4.1 that hidden strict information is a side-effect that can only occur in LHS-incoherent defeasible KBs.

We recall that the "right-to-left" direction of the proof for Theorem 1 (our reduction of exceptionality to classical entailment) on Page 143 holds for *general* $\langle \mathcal{T}, \mathcal{D} \rangle$'s (not just LHS-coherent ones) and therefore, the reduction stated in Theorem 1 is *sound* for exceptionality w.r.t. general $\langle \mathcal{T}, \mathcal{D} \rangle$'s but it is *incomplete* for these cases. That is, the "left-to-right" direction does not hold for general $\langle \mathcal{T}, \mathcal{D} \rangle$'s.

This is the reason why we cannot, in general, use our reduction to pinpoint *all* the exceptions in a LHS-incoherent KB (using Procedure `ComputeRankingA` on Page 150). However, we will show that, by performing some additional operations in Procedure `ComputeRankingA`, and using the same classical re-

duction we have presented, we can still pinpoint *all* the exceptions in such KBs. Before we give this complete procedure, we present an example to show the incompleteness of our reduction when applied to LHS-incoherent KBs. Consider the following example adapted from Proposition 2 on Page 141.

Example 18 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$*:*

$$\left\langle \left\{ \begin{array}{lcl} \mathsf{Dessert} & \sqsubseteq & \neg\mathsf{PizzaTopping}, \\ \mathsf{Fruit} & \sqsubseteq & \mathsf{Dessert} \end{array} \right\}, \left\{ \begin{array}{lcl} \mathsf{Fruit} & \precsim & \mathsf{PizzaTopping}, \\ \mathsf{DessertPizza} & \precsim & \exists \mathsf{hasTopping}.\mathsf{Fruit} \end{array} \right\} \right\rangle$$

□

In Example 18 on Page 155, notice that, from a semantic perspective, both Fruit and DessertPizza are totally exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$. This means we should obtain that both these concepts are exceptional using our reduction to classical entailment. That is, we should obtain both $\mathcal{T} \models \mathcal{C}_{\mathcal{D}'} \sqsubseteq \neg\mathsf{Fruit}$ and $\mathcal{T} \models \mathcal{C}_{\mathcal{D}'} \sqsubseteq \neg\mathsf{DessertPizza}$.

However, while we do have the former entailment, we do *not* have the latter, and we will construct a model for $\mathcal{T}$ to show this later in this example. Why do we get the former entailment, though? We observe that $\mathcal{T}$ effectively tells us that *all* Fruit objects are *not* PizzaTopping objects. On the other hand, the first statement in $\mathcal{D}$ tells us that *typical* Fruit objects are PizzaTopping objects. Clearly this is an incompatible situation and the result is that we will not find Fruit objects anywhere in any ranked model for our knowledge.

Therefore, we notice that it would be equivalent to replace $\mathsf{Fruit} \precsim \mathsf{PizzaTopping}$ with $\mathsf{Fruit} \sqsubseteq \mathsf{PizzaTopping}$ (i.e., with its strict counterpart). This effectively means we can remove the former statement from $\mathcal{D}$ and add its classical counterpart (the latter statement) to $\mathcal{T}$. In other words, $\mathsf{Fruit} \precsim \mathsf{PizzaTopping}$ is actually *hiding* its strict behaviour, or more eloquently, its strict behaviour is not "visible" to our reduction.

Consider the following model $\mathcal{I}$ for $\mathcal{T}$. $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ s.t. $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathsf{Dessert}^{\mathcal{I}} = \{b\}$, $\mathsf{PizzaTopping}^{\mathcal{I}} = \{a\}$, $\mathsf{Fruit}^{\mathcal{I}} = \{b\}$, $\mathsf{DessertPizza}^{\mathcal{I}} = \{a\}$ and $\mathsf{hasTopping}^{\mathcal{I}} = \{(a, b)\}$. It is easy to see that $a \in \mathcal{C}^{\mathcal{I}}_{\mathcal{D}'}$ and therefore $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}'} \sqsubseteq \neg\mathsf{DessertPizza}$.

Notice also that because of the non-globalisation of the constraint $\mathsf{Fruit} \mathrel{\tilde{\sqsubseteq}} \mathsf{PizzaTopping}$, Fruit objects are allowed to exist in the neighbourhood of a. That is, b is allowed to belong to $\mathsf{Fruit}^{\mathcal{I}}$. The end result is that the defeasibility of $\mathsf{Fruit} \mathrel{\tilde{\sqsubseteq}} \mathsf{PizzaTopping}$ actually "masks" the total exceptionality of DessertPizza.

Example 18 on Page 155 shows the incompleteness of our reduction of exceptionality to classical entailment for general $\langle \mathcal{T}, \mathcal{D} \rangle$'s. Unfortunately, there seems to be no such reduction that is declarative. Fortunately, there is an *operational* approach to cater for these cases using our existing reduction for LHS-coherent $\langle \mathcal{T}, \mathcal{D} \rangle$'s.

To convey the intuition of this procedure, consider the KB in Example 18 on Page 155. $\mathcal{T} \models \mathcal{C}_{\mathcal{D}'} \sqsubseteq \neg\mathsf{Fruit}$ but $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}'} \sqsubseteq \neg\mathsf{DessertPizza}$ (even though DessertPizza is totally exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$). We also observed that it is equivalent to remove $\mathsf{Fruit} \mathrel{\tilde{\sqsubseteq}} \mathsf{PizzaTopping}$ from $\mathcal{D}$ and add its classical counterpart to $\mathcal{T}$ (it is strict information parading as defeasible information). The resulting KB, say $\langle \mathcal{T}^*, \mathcal{D}^* \rangle$, would be:

$$\left\langle \left\{ \begin{array}{lcl} \mathsf{Dessert} & \sqsubseteq & \neg\mathsf{PizzaTopping}, \\ \mathsf{Fruit} & \sqsubseteq & \mathsf{Dessert}, \\ \mathsf{Fruit} & \sqsubseteq & \mathsf{PizzaTopping} \end{array} \right\}, \left\{ \begin{array}{lcl} \mathsf{DessertPizza} & \mathrel{\tilde{\sqsubseteq}} & \exists\mathsf{hasTopping.Fruit} \end{array} \right\} \right\rangle$$

It is straightforward to see that $\mathsf{Fruit} \sqsubseteq \bot$ follows classically from $\mathcal{T}^*$. Since strict information must be satisfied globally, it implies that, even after additionally considering the defeasible information in $\mathcal{D}^*$, there will not be any Fruit objects in any ranked model for the composite KB.

Now, it is interesting to observe that if we apply our reduction to check if DessertPizza is exceptional w.r.t. this modified KB (where all the hidden strict information is filtered into $\mathcal{T}$), we finally obtain $\mathcal{T}^* \models \mathcal{C}_{\mathcal{D}^*} \sqsubseteq \neg\mathsf{DessertPizza}$.

This behaviour tells us informally that we can apply our exceptionality reduction recursively on a defeasible KB to filter the hidden strict information into the TBox. It can be shown that eventually a point is reached where the incrementally modified defeasible KB becomes LHS-coherent, in which case

the procedure reduces to the same behaviour as Procedure `ComputeRankingA` on Page 150. Pseudocode of our general ranking procedure is given in Procedure `ComputeRankingB` on Page 157.

Procedure `ComputeRankingB`($\langle \mathcal{T}, \mathcal{D} \rangle$)

Input: A defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$.

Output: $\langle \mathcal{T}^*, \mathcal{D}^* \rangle$ (version of $\langle \mathcal{T}, \mathcal{D} \rangle$ in which all hidden strict information in $\mathcal{D}$ is moved to $\mathcal{T}$) and the ranking $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$.

1 $\mathcal{T}^* := \mathcal{T}$;
2 $\mathcal{D}^* := \mathcal{D}$;
3 $R := \emptyset$;
4 **repeat**
5 $i := 0$;
6 $\mathcal{E}_0 := \mathcal{D}^*$;
7 $\mathcal{E}_1 := \texttt{Exceptional}(\mathcal{T}^*, \mathcal{E}_0)$;
8 **while** $\mathcal{E}_{i+1} \neq \mathcal{E}_i$ **do**
9 $i := i + 1$;
10 $\mathcal{E}_{i+1} := \texttt{Exceptional}(\mathcal{T}^*, \mathcal{E}_i)$;
11 $\mathcal{D}^*_\infty := \mathcal{E}_i$;
12 $\mathcal{D}^* := \mathcal{D}^* \setminus \mathcal{D}^*_\infty$;
13 $\mathcal{T}^* := \mathcal{T}^* \cup \{C \sqsubseteq D \mid C \,\underset{\sim}{\sqsubset}\, D \in \mathcal{D}^*_\infty\}$;
14 **until** $\mathcal{D}^*_\infty = \emptyset$;
15 **for** $j = 1$ *to* i **do**
16 $\mathcal{D}_{j-1} := \mathcal{E}_{j-1} \setminus \mathcal{E}_j$;
17 $R := R \cup \{\mathcal{D}_{j-1}\}$;
18 **return** $\langle \mathcal{T}^*, \mathcal{D}^* \rangle, R$;

Procedure `ComputeRankingB` on Page 157 differs from the more basic Procedure `ComputeRankingA` on Page 150 mainly at Line 8 and Lines 11 – 13. Line 8 introduces a new termination condition for the computation of the

exceptionality subset sequence. Recall that in Procedure `ComputeRankingA` (at Line 5) the sequence terminates when none of the axioms in the given set are exceptional. I.e., the next set in the sequence will be empty. However, when the defeasible KB is LHS-incoherent then there is another possible termination condition. I.e., that *all* the axioms in a given set are exceptional.

A very simple example is the set $\{C \sqsubseteq D, C \sqsubseteq \neg D\}$. That is, if Procedure `ComputeRankingB` on Page 157 encounters such a set, it clearly denotes a fixed point (an indication that all axioms in this set are totally exceptional). When such a point is reached, we remove these hidden strict axioms from $\mathcal{D}$ (Line 12) and add their strict counterparts to $\mathcal{T}$ (Line 13). We have to repeatedly execute the exceptionality subset sequence procedure (Lines 4-14) whenever we have to transfer hidden strict information to the TBox. Eventually, the KB becomes LHS-coherent which means it does not contain totally exceptional axioms (Line 14).

After all hidden strict axioms are filtered out of $\mathcal{D}$, the algorithm works the same as Procedure `ComputeRankingA` on Page 150. We return the ranking as well as the modified defeasible KB (with the hidden strict axioms now in $\mathcal{T}$). It is easy to verify that executing Procedure `ComputeRankingB` (on Page 157) on $\langle \mathcal{T}, \mathcal{D} \rangle$ from Example 18 (on Page 155) gives $R = \emptyset$ and $\langle \mathcal{T}^*, \mathcal{D}^* \rangle$:

$$\left\langle \left\{ \begin{array}{lcl} \mathsf{Dessert} & \sqsubseteq & \neg\mathsf{PizzaTopping}, \\ \mathsf{Fruit} & \sqsubseteq & \mathsf{Dessert}, \\ \mathsf{Fruit} & \sqsubseteq & \mathsf{PizzaTopping}, \\ \mathsf{DessertPizza} & \sqsubseteq & \exists\mathsf{hasTopping.Fruit} \end{array} \right\}, \emptyset \right\rangle$$

We have to show that Procedure `ComputeRankingB` on Page 157 terminates. Essentially, we have to show that the while loop and the repeat loop terminate. It is fairly straightforward to see that there are only two cases: (1) $\langle \mathcal{T}, \mathcal{D} \rangle$ is LHS-coherent, in which case termination of the while loop follows immediately from our arguments for Procedure `ComputeRankingA` on Page 150. In addition, from the same arguments we know that termination

happens when $\mathcal{E}_i = \emptyset$ for some i. This means that $\mathcal{D}^*_\infty = \emptyset$ (Line 11) and then we terminate the repeat loop because of Line 14.

(2) the other case is when $\langle \mathcal{T}, \mathcal{D} \rangle$ is LHS-incoherent. If there is no i for which $\mathcal{E}_i = \emptyset$ it means that the exceptionality subsets have stopped decreasing in size so that $\mathcal{E}_i = \mathcal{E}_{i+1} \neq \emptyset$ for some i and therefore the while loop terminates. To show that the repeat loop terminates for this case, observe that the fixed point axioms are removed from $\mathcal{D}^*$ (Line 12) and we recurse on this smaller $\mathcal{D}^*$ (Line 6). Since $\mathcal{D}$ is finite we have to reach the point where $\mathcal{E}_j$ is empty for some $j > i$.

Observe that the additional operations in Procedure `ComputeRankingB` on Page 157 are only introduced to handle hidden strict information. Hidden strict information, in turn, is ranked unsatisfiable information. We notice that, by definition, ranked unsatisfiable information has infinite rank (see Definitions 10 and 12). This is the motivation for the choice of symbol $\mathcal{D}^*_\infty$ representing the set of hidden strict subsumptions.

Note that Procedure `ComputeRankingB` on Page 157 transfers all such information to the TBox. From a semantic perspective, it is clear that such behaviour cannot influence the rank of ranked *satisfiable* information in the KB (because $C \,\underset{\sim}{\sqsubseteq}\, \bot$ is ranked equivalent to $C \sqsubseteq \bot$). Finally, we observe that, apart from the infinite rank information, Procedure `ComputeRankingB` on Page 157 treats the remainder information in the same way as Procedure `ComputeRankingA` on Page 150.

The arguments expressed above give credibility to the claim that the soundness of Procedure `ComputeRankingA` on Page 150 transfers over to the context of Procedure `ComputeRankingB` on Page 157 as well. However, this can also be formally demonstrated by the following lemma.

Lemma 8 (Soundness of Procedure `ComputeRankingB`) *Given a LHS-incoherent defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, for each $C \,\underset{\sim}{\sqsubseteq}\, D \in \mathcal{D}$, $C \,\underset{\sim}{\sqsubseteq}\, D \in \mathcal{D}^*_\infty$ (for some $\mathcal{D}^*_\infty$ during iteration of the repeat loop of Procedure `ComputeRankingB` on Page 157) if and only if $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \,\underset{\sim}{\sqsubseteq}\, \bot$.*

Proof: " $\Longrightarrow$ " We have to show that if $C \,\underset{\sim}{\sqsubseteq}\, D \in \mathcal{D}^*_\infty$ for some $\mathcal{D}^*_\infty$ during ex-

ecution of Procedure `ComputeRankingB` on Page 157, then $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \sqsubseteq \bot$. It is clear that $\mathcal{D}^*_\infty$ is the fixed point set in which all axioms are exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ (Line 11). If we take the contrapositive of Lemma 6 on Page 152 we find that $\mathcal{D}^*_\infty$ is LHS-incoherent and therefore $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \sqsubseteq \bot$.
" $\Longleftarrow$ " We have to show that for each $C \sqsubseteq D \in \mathcal{D}$, if $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \sqsubseteq \bot$ then $C \sqsubseteq D \in \mathcal{D}^*_\infty$ for some $\mathcal{D}^*_\infty$ during execution of Procedure `ComputeRankingB` on Page 157. Informally, what we have to show is that, even though we know that our reduction is incomplete for general $\langle \mathcal{T}, \mathcal{D} \rangle$'s (see Example 18 on Page 155), it will eventually still capture *all* the hidden strict information (by recursive application of Lines $4-14$). If this was not the case, it entails that there is a $C \sqsubseteq D \in \mathcal{D}$ s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \sqsubseteq \bot$ but for each $\mathcal{D}^*_\infty$ in the execution of Procedure `ComputeRankingB` on Page 157, $C \sqsubseteq D \notin \mathcal{D}^*_\infty$. It is easy to see then that upon termination of Procedure `ComputeRankingB` we have a $\langle \mathcal{T}^*, \mathcal{D}^* \rangle$ s.t. for each $C \sqsubseteq D \in \mathcal{D}^*$ s.t. $\langle \mathcal{T}^*, \mathcal{D}^* \rangle \models_r C \sqsubseteq \bot$, $\mathcal{T}^* \not\models \mathcal{C}_{\mathcal{D}^*} \sqsubseteq \neg C$ (our reduction fails to recognise the exceptionality of C). We will show that this is impossible.

I.e., we pick a $C \sqsubseteq D \in \mathcal{D}^*$ s.t. $\langle \mathcal{T}^*, \mathcal{D}^* \rangle \models_r C \sqsubseteq \bot$. We know that $\mathcal{T}^* \not\models \mathcal{C}_{\mathcal{D}^*} \sqsubseteq \neg C$ (our reduction fails to recognise the exceptionality of C). But Procedure `ComputeRankingB` on Page 157 also partitions the KB into a ranking $\mathcal{D}_0, \ldots, \mathcal{D}_n$. Therefore, because our reduction has "missed" the total exceptionality of C, there will be some $0 \leq i \leq n$ s.t. $C \sqsubseteq D \in \mathcal{D}_i$, $\mathcal{T}^* \cup \mathcal{D}_i \models_r \top \sqsubseteq \neg C$ and for each $X \sqsubseteq Y \in \mathcal{D}_i$, $\mathcal{T}^* \not\models \mathcal{C}_{\mathcal{D}_i} \sqsubseteq \neg X$. We pick a model $\mathcal{I}_X$ for $\mathcal{T}^*$ for each $X \sqsubseteq Y \in \mathcal{D}_i$ s.t. there is an $x \in (\mathcal{C}_{\mathcal{D}_i} \sqcap X)^{\mathcal{I}_X}$. We take the disjoint union (Definition 26 on Page 131) of these models to obtain $\mathcal{I} = \biguplus \mathcal{I}_X$ for each $X \sqsubseteq Y \in \mathcal{D}_i$. By Lemma 1 on Page 132 we know that $\mathcal{I}$ is a model for $\mathcal{T}^*$.
Now, let $\mathcal{R} = \langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}}, \prec_{\mathcal{R}} \rangle$ be a ranked interpretation s.t. $\Delta^{\mathcal{R}} = \Delta^{\mathcal{I}}$, $\cdot^{\mathcal{R}} = \cdot^{\mathcal{I}}$ and $\prec_{\mathcal{R}}$ is defined s.t. for each $x \in \mathcal{C}^{\mathcal{R}}_{\mathcal{D}_i}$, $rk_{\mathcal{R}}(x) = 0$, and, for each $y \notin \mathcal{C}^{\mathcal{R}}_{\mathcal{D}_i}$, $rk_{\mathcal{R}}(y) = 1$. We have to show that $\mathcal{R} \Vdash \mathcal{T}^* \cup \mathcal{D}_i$. It is clear that $\mathcal{R} \Vdash \mathcal{T}^*$ because $\Delta^{\mathcal{R}} = \Delta^{\mathcal{I}}$, $\cdot^{\mathcal{R}} = \cdot^{\mathcal{I}}$ and $\mathcal{I} \Vdash \mathcal{T}^*$. Assume that $\mathcal{R} \not\Vdash \mathcal{D}_i$. This implies that there is a $X \sqsubseteq Y \in \mathcal{D}_i$ s.t. $\mathcal{R} \not\Vdash X \sqsubseteq Y$. This means there is a

$z \in (X \sqcap \neg Y)^{\mathcal{R}}$ s.t. there is no $z' \prec_{\mathcal{R}} z$ s.t. $z' \in X^{\mathcal{R}}$. There are two cases:
Case 1: $rk_{\mathcal{R}}(z) = 0$. This case is clearly impossible because $z \in \mathcal{C}^{\mathcal{R}}_{\mathcal{D}_i}$ which implies that $z \in (\neg X \sqcup Y)^{\mathcal{R}}$.
Case 2: $rk_{\mathcal{R}}(z) = 1$. This implies that there is no $z' \in X^{\mathcal{R}}$ s.t. $rk_{\mathcal{R}}(z') = 0$. But by definition of $\mathcal{R}$ (and $\mathcal{I}$), there exists a $y \in (\mathcal{C}^{\mathcal{R}}_{\mathcal{D}_i} \sqcap X)^{\mathcal{R}}$. By definition of $\prec_{\mathcal{R}}$, $rk_{\mathcal{R}}(y) = 0$. This is a contradiction with z'.
Therefore, we have shown that $\mathcal{R} \Vdash \mathcal{T}^* \cup \mathcal{D}_i$. But we also know that there is an $y \in (\mathcal{C}^{\mathcal{R}}_{\mathcal{D}_i} \sqcap C)^{\mathcal{R}}$ s.t. $rk_{\mathcal{R}}(y) = 0$. This is clearly in contradiction with our earlier finding that $\mathcal{T}^* \cup \mathcal{D}_i \models_r \top \subseteq \neg C$. Therefore, we have proven that if $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \subseteq \bot$ then $C \subseteq D \in \mathcal{D}^*_\infty$ for some $\mathcal{D}^*_\infty$ during the execution of Procedure `ComputeRankingB` on Page 157. □

We now discuss the phenomenon of hidden strict information from the user perspective. It must be stressed that our view is that hidden strict information is a symptom of poor knowledge engineering choices. That is, we believe it is a result of ontology modelling errors. We note that such axioms are totally exceptional, which is equivalent to saying that their antecedents are ranked unsatisfiable. Preferential unsatisfiability itself is analogously defined to classical DL unsatisfiability. There is a general consensus in the literature that classical DL unsatisfiability is, in most cases, an erroneous situation and the result of incorrect ontology design choices. Therefore, this argument can be inherited by the case of ranked unsatisfiability as well.

In fact, we argue that ranked unsatisfiability is even stronger evidence for poor design choices than classical unsatisfiability. Informally, this can be explained by observing that, while there are multifarious causes for classical unsatisfiability, the reason for ranked unsatisfiability is comparatively more focused. That is, it is a caused by conflicts between defeasible information of the *same* priority or degree of exceptionality. We give some examples:

Example 19 *Consider the following classical KB* K*:*

$$\left\{\begin{array}{lcl} \mathsf{Student} & \sqsubseteq & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} & \sqsubseteq & \mathsf{Student}, \\ \mathsf{EmployedStudent} & \sqsubseteq & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \sqsubseteq & \neg(\exists \mathsf{receives.TaxInvoice}) \end{array}\right\}$$

□

It is clear from the KB in Example 19 on Page 162 that $\mathsf{EmployedStudent}$ and $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$ are both classically unsatisfiable w.r.t. K. That is, no objects belonging to these concepts can occur in any standard DL model for K. Translating all the axioms in K to their defeasible counterparts we obtain the defeasible KB K':

$$\left\{\begin{array}{lcl} \mathsf{Student} & \mathrel{\tilde{\sqsubseteq}} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} & \mathrel{\tilde{\sqsubseteq}} & \mathsf{Student}, \\ \mathsf{EmployedStudent} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \mathrel{\tilde{\sqsubseteq}} & \neg(\exists \mathsf{receives.TaxInvoice}) \end{array}\right\}$$

We note that K' is LHS-coherent, which means that neither $\mathsf{EmployedStudent}$ nor $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$ are ranked unsatisfiable w.r.t. K'. However, both these concepts are normally exceptional w.r.t. K'. Even though they are exceptional, objects that belong to these concepts can be reconciled in a ranked model for K'. We have the freedom to spread these objects across the levels of the ranked model if they cannot peacefully co-exist on the same level. The following KBs illustrate a similar situation to Example 19 on Page 162:

$$\mathsf{K} = \left\{\begin{array}{lcl} \top & \sqsubseteq & \neg\mathsf{D}, \\ \mathsf{C} & \sqsubseteq & \exists \mathsf{R.D} \end{array}\right\} \quad \mathsf{K}' = \left\{\begin{array}{lcl} \top & \mathrel{\tilde{\sqsubseteq}} & \neg\mathsf{D}, \\ \mathsf{C} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{R.D} \end{array}\right\}$$

Again we find that C and D are classically unsatisfiable w.r.t. K and, in contrast, neither is ranked unsatisfiable w.r.t. K'. In fact, C is not even

normally exceptional w.r.t. K'. The semantics of ranked models allows the power to reconcile the defeasible axioms by spreading C and D objects across the levels of the ranked models. As we shall see in the next three example KBs, ranked unsatisfiability is arrived at when we force incompatibility in the confines of a particular level of a ranked model:

$$\mathsf{K} = \left\{ \begin{array}{lll} \mathsf{C} & \sqsubseteq & \neg\mathsf{D}, \\ \mathsf{C} & \sqsubseteq & \mathsf{D} \end{array} \right\} \qquad \mathsf{K}' = \left\{ \begin{array}{lll} \mathsf{C} & \tilde{\sqsubseteq} & \neg\mathsf{D}, \\ \mathsf{C} & \tilde{\sqsubseteq} & \mathsf{D} \end{array} \right\}$$

Notice that in K we have a very crude manner of enforcing that C is classically unsatisfiable. That is, there is a *direct* conflict in the "definition" of the concept C given in K. It is not an *indirect* conflict in the sense that there is a *sub-concept* of C whose definition conflicts with that of C. Therefore, we find that this carries over to the defeasible translation K' because here C is ranked unsatisfiable. If we allow that there be strict axioms in our defeasible translation then we can obtain more interesting examples of this kind of conflict.

$$\mathsf{K} = \left\{ \begin{array}{lll} \mathsf{Dessert} & \sqsubseteq & \neg\mathsf{PizzaTopping}, \\ \mathsf{Fruit} & \sqsubseteq & \mathsf{Dessert}, \\ \mathsf{Fruit} & \sqsubseteq & \mathsf{PizzaTopping} \end{array} \right\} \qquad \mathsf{K}' = \left\{ \begin{array}{lll} \mathsf{Dessert} & \sqsubseteq & \neg\mathsf{PizzaTopping}, \\ \mathsf{Fruit} & \tilde{\sqsubseteq} & \mathsf{Dessert}, \\ \mathsf{Fruit} & \tilde{\sqsubseteq} & \mathsf{PizzaTopping} \end{array} \right\}$$

In the KBs above we allow the defeasible translation K' to retain a strict axiom - $\mathsf{Dessert} \sqsubseteq \neg\mathsf{PizzaTopping}$. We find that Fruit is classically unsatisfiable w.r.t. K and also ranked unsatisfiable w.r.t. K'. There is a conflict in the definition of Fruit, i.e., that Fruit is enforced to belong to $\mathsf{PizzaTopping}$ and its complement. Fruit is enforced to belong to $\neg\mathsf{PizzaTopping}$ because we know that $\mathsf{Fruit} \tilde{\sqsubseteq} \mathsf{Dessert}$, $\mathsf{Dessert} \sqsubseteq \neg\mathsf{PizzaTopping}$ and by application of the Right Weakening KLM postulate we arrive at $\mathsf{Fruit} \tilde{\sqsubseteq} \neg\mathsf{PizzaTopping}$.

We can even extend the above example to include roles, reiterating that ranked unsatisfiability can be propagated through roles. We add the axiom $\mathsf{DessertPizza} \sqsubseteq \exists\mathsf{hasTopping}.\mathsf{Fruit}$ to the above KBs to obtain.

$$\mathsf{K} = \left\{ \begin{array}{lcl} \mathsf{Dessert} & \sqsubseteq & \neg\mathsf{PizzaTopping}, \\ \mathsf{Fruit} & \sqsubseteq & \mathsf{Dessert}, \\ \mathsf{Fruit} & \sqsubseteq & \mathsf{PizzaTopping}, \\ \mathsf{DessertPizza} & \sqsubseteq & \exists\mathsf{hasTopping.Fruit} \end{array} \right\} \mathsf{K}' = \left\{ \begin{array}{lcl} \mathsf{Dessert} & \sqsubseteq & \neg\mathsf{PizzaTopping}, \\ \mathsf{Fruit} & \mathrel{\tilde{\sqsubseteq}} & \mathsf{Dessert}, \\ \mathsf{Fruit} & \mathrel{\tilde{\sqsubseteq}} & \mathsf{PizzaTopping}, \\ \mathsf{DessertPizza} & \mathrel{\tilde{\sqsubseteq}} & \exists\mathsf{hasTopping.Fruit} \end{array} \right\}$$

The classical unsatisfiability of Fruit (w.r.t. K) is retained in the above example. In addition we find that DessertPizza is also unsatisfiable w.r.t. K. When moving over to the defeasible translation K′, we find that both Fruit and DessertPizza are also ranked unsatisfiable. That is, knowing that we cannot realise Fruit objects in a ranked model of the knowledge, if we enforce that DessertPizza objects have relations to Fruit objects, then DessertPizza objects cannot exist either. This behaviour closely resembles the behaviour in classical DLs (i.e., the classical unsatisfiability of a concept can propagate through roles).

To summarise our discussion about the logical merit of hidden strict information, we maintain that it is an indication of errors in the ontology engineering process. In the framework of ranked model semantics, we note that it corresponds to conflicts in defeasible subsumptions of the same priority and, given an example such as $\{C \mathrel{\tilde{\sqsubseteq}} D, C \mathrel{\tilde{\sqsubseteq}} \neg D\}$, the framework does not attempt to choose between accepting $C \mathrel{\tilde{\sqsubseteq}} D$ and $C \mathrel{\tilde{\sqsubseteq}} \neg D$ from an inference perspective. Rather, the framework treats this situation as a form of logical incoherence (it is the same as accepting $\{C \sqsubseteq D, C \sqsubseteq \neg D\}$) and leaves this to be debugged by the ontology engineer if desired. This philosophy aligns with other approaches such as the one by Bonatti et al [27, Example 1] for overriding in DLs (see Section 2.9).

To conclude this section on the ranking of a defeasible KB, we demonstrate two important properties of this ranking. The first is that the ranking respects the property of *specificity*, and the second is that it is possible for the ranking to be refined further by user input if desired. We will make the latter property more precise at the end of this section. For now we demonstrate that the ranking respects specificity. The proof statement is represented in the following lemma. Notice that this captures a more powerful version of specificity based on defeasible subsumption.

Lemma 9 (Ranking Respects Specificity) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a defeasible KB. Then, for each pair of defeasible subsumptions $C_1 \sqsubseteq \!\!\!\sim D_1, C_2 \sqsubseteq \!\!\!\sim D_2 \in \mathcal{D}$, if $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C_1 \sqsubseteq \!\!\!\sim C_2$ and $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r C_2 \sqsubseteq \!\!\!\sim C_1$ then $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C_1) \geq rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C_2)$.*

Proof: Suppose there exists a pair of subsumptions $C_1 \sqsubseteq \!\!\!\sim D_1, C_2 \sqsubseteq \!\!\!\sim D_2 \in \mathcal{D}$ s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C_1 \sqsubseteq \!\!\!\sim C_2$, $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r C_2 \sqsubseteq \!\!\!\sim C_1$ and, in contrast to the consequent of Lemma 9 on Page 165, $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C_1) < rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C_2)$. We pick a ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. $rk_{\mathcal{R}}(C_1) = rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C_1)$. Notice that picking an element $x \in \min_{\prec_{\mathcal{R}}}(C_1^{\mathcal{R}})$ we know that $x \in C_2^{\mathcal{R}}$ ($\langle \mathcal{T}, \mathcal{D} \rangle \models_r C_1 \sqsubseteq \!\!\!\sim C_2$). But then it follows from this that $rk_{\mathcal{R}}(C_2) \leq rk_{\mathcal{R}}(C_1)$ which means that $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C_2) \leq rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C_1)$ which contradicts with $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C_1) < rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(C_2)$. □

Observe that Lemma 9 on Page 165 defines specificity of knowledge w.r.t. defeasible subsumption as well (as opposed to purely strict information like Bonatti et al. [27, Page 11] do). Thus, a desirable property of the ranking is that it will not only satisfy specificity induced by defeasible subsumption, but also specificity induced by strict information (strict subsumption is logically stronger than defeasible subsumption). In other words, $\mathcal{T} \models C \sqsubseteq D \implies \langle \mathcal{T}, \mathcal{D} \rangle \models_r C \sqsubseteq D \implies \langle \mathcal{T}, \mathcal{D} \rangle \models_r C \sqsubseteq \!\!\!\sim D$. Consider the following examples.

Example 20 *Consider the following defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$:*

$$\left\langle \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\}, \left\{ \mathsf{Student} \sqsubseteq \!\!\!\sim \neg(\exists \mathsf{receives.TaxInvoice}) \right\} \right\rangle$$ □

In Example 20 on Page 165, employed students are more specific than general students (induced by the strict subsumption in $\mathcal{T}$). Of course, the ranking respects this specificity relationship because $rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(\mathsf{EmployedStudent}) = rk_{\langle \mathcal{T}, \mathcal{D} \rangle_s}(\mathsf{Student})$. Note that the reason why they can share the same rank is that there is no information in $\langle \mathcal{T}, \mathcal{D} \rangle$ which forces one to consider any of the concepts as exceptional. Now, if we add such information to our knowledge we have $\langle \mathcal{T}, \mathcal{D} \rangle$:

$$\left\langle \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\}, \left\{ \begin{array}{lll} \mathsf{Student} & \sqsubseteq \!\!\!\sim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} & \sqsubseteq \!\!\!\sim & \exists \mathsf{receives.TaxInvoice} \end{array} \right\} \right\rangle$$

Again, employed students are more specific than general students (induced by the strict subsumption in $\mathcal{T}$). In addition, notice that $\mathsf{EmployedStudent}$ is exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ so the ranking ensures that $rk_{\langle \mathcal{T},\mathcal{D} \rangle_s}(\mathsf{EmployedStudent}) > rk_{\langle \mathcal{T},\mathcal{D} \rangle_s}(\mathsf{Student})$. Now, consider the case where our strict subsumption is represented as a defeasible subsumption:

$$\left\langle \emptyset, \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \sqsubseteq\!\!\!\sim & \mathsf{Student}, \\ \mathsf{Student} & \sqsubseteq\!\!\!\sim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} & \sqsubseteq\!\!\!\sim & \exists \mathsf{receives.TaxInvoice} \end{array} \right\} \right\rangle$$

Again, the ranking preserves $rk_{\langle \mathcal{T},\mathcal{D} \rangle_s}(\mathsf{EmployedStudent}) > rk_{\langle \mathcal{T},\mathcal{D} \rangle_s}(\mathsf{Student})$ because it respects specificity induced by defeasible subsumption. This is arguably quite intuitive because there is no information precluding us from doing so. However, it is perhaps not intuitive for $\mathsf{EmployedStudent} \sqsubseteq\!\!\!\sim \mathsf{Student}$ to be represented defeasibly. One would imagine that all employed students are students. That is, it does not make sense to weaken such a statement to say that just the typical employed students are students.

We now demonstrate that, if it is desirable to the user, the ranking can be modified (under some restriction), while still preserving properties such as respect for specificity. Consider the following defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ where:

$$\mathcal{T} = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \sqsubseteq & \mathsf{Student}, \\ \mathsf{SelfSponsoredStudent} & \sqsubseteq & \mathsf{Student} \end{array} \right\},$$

$$\mathcal{D} = \left\{ \begin{array}{lcl} \mathsf{Student} & \sqsubseteq\!\!\!\sim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \sqsubseteq\!\!\!\sim & \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{Student} & \sqsubseteq\!\!\!\sim & \neg \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \sqsubseteq\!\!\!\sim & \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \sqsubseteq\!\!\!\sim & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \sqsubseteq\!\!\!\sim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{worksFor.University} & \sqsubseteq\!\!\!\sim & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\}$$

Applying the ranking procedure (Procedure `ComputeRankingB` on Page 157) to $\langle \mathcal{T}, \mathcal{D} \rangle$ we obtain the ranking:

$$\mathcal{D}_0 = \left\{ \begin{array}{lcl} \mathsf{Student} & \sqsubseteq & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \sqsubseteq & \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{Student} & \sqsubseteq & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\},$$

$$\mathcal{D}_1 = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \sqsubseteq & \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \sqsubseteq & \exists \mathsf{receives.TaxInvoice} \end{array} \right\},$$

$$\mathcal{D}_2 = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \sqsubseteq & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{worksFor.University} & \sqsubseteq & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\}$$

$\{\mathcal{D}_0, \mathcal{D}_1, \mathcal{D}_2\}$ represents the exceptionality ranking of $\langle \mathcal{T}, \mathcal{D} \rangle$. It is important to understand that this ranking is induced by exceptionality (of the antecedent concepts of the subsumptions in $\mathcal{D}$). Preferential reasoning entailment regimes generally use this base ranking to aid in deciding entailment (details are presented in subsequent sections of this chapter). However, the ranking can be *refined* if desired by the user. Informally, this refinement is a re-assignment of ranking values to each sentence in $\mathcal{D}$. Of course, in order to guarantee the elegant mathematical properties of the intended entailment regime, there needs to be a restriction on how the ranking may be modified. Before we describe this restriction we define a *general refinement* of a ranking.

Definition 34 (General Ranking Refinement) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a defeasible KB and $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its ranking. Then, a* general refinement *for R is a set $R' = \{\mathcal{D}'_0, \ldots, \mathcal{D}'_m\}$ s.t. $\mathcal{D}_0 \cup \ldots \cup \mathcal{D}_n = \mathcal{D}'_0 \cup \ldots \cup \mathcal{D}'_m$ and $m \geq n$.*

Now we have to define the conditions under which a refinement of a ranking will still preserve the desirable properties of rational consequence relations (when used in the computation of inference).

Definition 35 (Safe Ranking Refinement)
Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a defeasible KB, $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its ranking and $R' = \{\mathcal{D}'_0, \ldots, \mathcal{D}'_m\}$ a general ranking refinement for R. Then, R' is a safe *ranking refinement for R if: for each pair of axioms $C_1 \sqsubseteq D_1 \in \mathcal{D}_i, \mathcal{D}'_k$ and $C_2 \sqsubseteq D_2 \in \mathcal{D}_j, \mathcal{D}'_l$ (where $0 \leq i, j \leq n$ and $0 \leq k, l \leq m$), if $i > j$ then $k > l$.*

Informally, Definition 35 on Page 167 says that a safe ranking refinement preserves the *relative* ranks of sentences that do not have the *same* rank. In other words, a meaningful safe refinement for R tries to re-assign ranks to sentences that are of the same rank. For example, a user may decide that the original ranking $\{\mathcal{D}_0, \mathcal{D}_1, \mathcal{D}_2\}$ in the above example is too coarse for her. That is, she may decide that employed students' tax paying property is more important than whether they are self sponsored or not. Therefore, she can "split" the rank $\mathcal{D}_1$ into two sub-ranks: $\{\mathsf{EmployedStudent} \sqsubseteq \mathsf{SelfSponsoredStudent}\}$ and the other (higher sub-rank) $\{\mathsf{EmployedStudent} \sqsubseteq \exists \mathsf{receives.TaxInvoice}\}$. The refined ranking R' would then be:

$$\mathcal{D}_0 = \left\{ \begin{array}{lll} \mathsf{Student} & \sqsubseteq & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \sqsubseteq & \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{Student} & \sqsubseteq & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\},$$

$$\mathcal{D}_1 = \left\{ \mathsf{EmployedStudent} \quad \sqsubseteq \quad \mathsf{SelfSponsoredStudent} \right\},$$

$$\mathcal{D}_2 = \left\{ \mathsf{EmployedStudent} \quad \sqsubseteq \quad \exists \mathsf{receives.TaxInvoice} \right\},$$

$$\mathcal{D}_3 = \left\{ \begin{array}{lll} \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \sqsubseteq & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{worksFor.University} & \sqsubseteq & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\}$$

Of course, similar refinements can be made for $\mathcal{D}_0$ and $\mathcal{D}_3$. Notice that such refinements will result in rankings that respect the exceptionality-induced ranking. In other words this means that, according to the original ranking, if a concept C_1 is more exceptional than a concept D_1, then in the refined ranking C_1 will still have a higher rank than D_1. In contrast, we may have the case where our original ranking stipulates that two concepts C_2 and D_2 have equal exceptionality (appear in the same rank), but in the refined ranking we are allowed to make a distinction between them (we can specify that C_2 has a higher rank than D_2 or vice versa). The important thing to observe is that this refinement is not motivated by exceptionality, but would be defined by other user-centered desiderata. We have thus given a presentation of the

major details and intuitions pertaining to the ranking of a defeasible KB. We reiterate that this ranking is the backbone of all entailment regimes, presented in this chapter, for deciding inference in the preferential context. Arguably the most important of these (at least from a theoretical perspective) is the most conservative rational consequence relation called Rational Closure which we discuss in the next section.

4.3 Rational Closure

It was shown by KLM that Rational Closure is the most conservative rational consequence relation. By "most conservative" we mean "gives back the fewest positive entailments", where a positive entailment is one of the form $\langle \mathcal{T}, \mathcal{D} \rangle \models_{rational} C \sqsubseteq D$ (as opposed to a negative entailment of the form $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_{rational} C \sqsubseteq D$). Given the semantic definition of Rational Closure (Definition 20 on Page 95), as well as a procedure for computing the rank of a concept w.r.t. a defeasible KB, it is straightforward to give an algorithmic characterisation for Rational Closure.

That is, given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and a query axiom $C \sqsubseteq D$, to decide if $C \sqsubseteq D$ is in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$ we need to compute the ranks of C and $C \sqcap \neg D$ w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$. Notice that this corresponds to identifying where to place C and $C \sqcap \neg D$ in the exceptionality ranking R for $\langle \mathcal{T}, \mathcal{D} \rangle$.

Then, from Definition 20 on Page 95, it is straightforward that $C \sqsubseteq D$ is in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$ if the rank of C is strictly less than the rank of $C \sqcap \neg D$ (informally, if the objects of $C \sqcap \neg D$ are more exceptional than objects of C). We start off by giving a pseudocode procedure for computing the rank of an arbitrary concept in a given ranking R for some defeasible KB. Recall that Procedure `ComputeRankingA` on Page 150 is used to compute the rank of LHS concepts of defeasible subsumptions in the KB. Also, recall that this procedure actually depends on Procedure `Exceptional` on Page 148 (which computes the exceptionality subset sequence). Hence, our procedure here is just a modification of Procedure `Exceptional` to compute the rank

of an *arbitrary* concept (one that does not necessarily appear in the KB).

Procedure `Rank`$(\langle \mathcal{T}, \mathcal{D} \rangle, R, C)$

Input: A LHS-coherent defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, its ranking $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ and a concept C.

Output: The natural number value representing the rank of C w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$.

```
1 i:=0;
2 while T |= C_{D_n ∪ ... ∪ D_i} ⊑ ¬C do
3     if i = n then
4         return ∞;
5     i := i + 1;
6 return i;
```

The critical thing to notice about Procedure `Rank` on Page 170 is that it only accepts LHS-coherent $\langle \mathcal{T}, \mathcal{D} \rangle$'s as input. The reason why we only need to consider LHS-coherent KBs is that, after the computation of the ranking for a *general* defeasible KB, all the hidden strict inclusions are moved to the TBox (see Procedure `ComputeRankingB` on Page 157). The modified KB (where all hidden strict information is moved to the TBox) is then returned by Procedure `ComputeRankingB` (see Line 18 of the procedure). This KB is, by definition, LHS-coherent and thus we only need to consider these KBs.

It is clear that the procedure terminates when $i = n$ (Line $3 - 4$). Notice that $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n} \sqsubseteq \neg C$ (Line 2) represents the case where C is totally exceptional. Since the procedure closely follows the ranking procedures in Section 4.2, the soundness (the computed rank of the concept corresponds with the semantic notion of the rank of a concept) follows from our arguments in that section. Now that we are able to compute the rank of a concept, the procedure to compute Rational Closure is a straightforward characterisation of its semantic definition.

Because Procedure `RationalClosureA` on Page 171 closely follows the se-

Procedure `RationalClosureA(`$\langle\mathcal{T},\mathcal{D}\rangle$`,`$R$`,`$\delta$`)`

Input: A LHS-coherent defeasible KB $\langle\mathcal{T},\mathcal{D}\rangle$, its ranking $R = \{\mathcal{D}_0,\ldots,\mathcal{D}_n\}$ and a defeasible subsumption $\delta = C \mathrel{\sqsubset\!\!\sim} D$.

Output: **true** if δ is in the Rational Closure of $\langle\mathcal{T},\mathcal{D}\rangle$, **false** otherwise.

```
1 if Rank(⟨T,D⟩, R, C) <
  Rank(⟨T,D⟩, R, C ⊓ ¬D) or Rank(⟨T,D⟩, R, C) = ∞ then
2     return true;
3 return false;
```

mantic definition of Rational Closure, and the procedural and semantic definitions for the rank of a concept correspond (see Section 4.2), it is clear that the procedure is sound. Here's an example to illustrate its behaviour:

Example 21 *Consider the following defeasible KB* $\langle\mathcal{T},\mathcal{D}\rangle$*:*

$$\left\langle \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\}, \left\{ \begin{array}{lll} \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{receives.TaxInvoice} \end{array} \right\} \right\rangle \quad \square$$

The ranking R for $\langle\mathcal{T},\mathcal{D}\rangle$ in Example 21 on Page 171 is the sequence of subsets of $\mathcal{D}$:

$$\begin{array}{lll} \mathcal{D}_0 = \{\mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice})\}, \\ \mathcal{D}_1 = \{\mathsf{EmployedStudent} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{receives.TaxInvoice}\} \end{array}$$

Suppose we want to verify if the axioms $\mathsf{Student} \mathrel{\sqsubset\!\!\sim} \neg(\exists \mathsf{receives.TaxInvoice})$, $\mathsf{EmployedStudent} \mathrel{\sqsubset\!\!\sim} \exists \mathsf{receives.TaxInvoice}$ and $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \mathrel{\sqsubset\!\!\sim} \neg(\exists \mathsf{receives.TaxInvoice})$ are in the Rational Closure for $\langle\mathcal{T},\mathcal{D}\rangle$. It is straightforward to determine:

$$\begin{array}{lcl} \mathtt{Rank}(\langle\mathcal{T},\mathcal{D}\rangle, R, \mathsf{Student}) & = & 0, \\ \mathtt{Rank}(\langle\mathcal{T},\mathcal{D}\rangle, R, \mathsf{EmployedStudent}) & = & 1, \\ \mathtt{Rank}(\langle\mathcal{T},\mathcal{D}\rangle, R, \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top) & = & 1 \end{array}$$

In addition, we can determine that:

$$\begin{aligned}
\texttt{Rank}(\langle\mathcal{T},\mathcal{D}\rangle, R, \mathsf{Student} \sqcap \exists \mathsf{receives.TaxInvoice}) &= 1,\\
\texttt{Rank}(\langle\mathcal{T},\mathcal{D}\rangle, R, \mathsf{EmployedStudent} \sqcap \neg(\exists \mathsf{receives.TaxInvoice})) &= \infty,\\
\texttt{Rank}(\langle\mathcal{T},\mathcal{D}\rangle, R, \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \sqcap \exists \mathsf{receives.TaxInvoice}) &= 1
\end{aligned}$$

Therefore we can conclude that axioms $\mathsf{Student} \mathrel{\tilde{\sqsubset}} \neg(\exists \mathsf{receives.TaxInvoice})$ and $\mathsf{EmployedStudent} \mathrel{\tilde{\sqsubset}} \exists \mathsf{receives.TaxInvoice}$ are in the Rational Closure of $\langle\mathcal{T},\mathcal{D}\rangle$, while $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \mathrel{\tilde{\sqsubset}} \neg(\exists \mathsf{receives.TaxInvoice})$ is not.

While Procedure `RationalClosureA` on Page 171 is straightforward and amenable to practical implementation, it is not the most efficient algorithmic characterisation for Rational Closure. We will give an alternative characterisation which, from a practical perspective, is computationally less intensive. However, before we give this procedure we must introduce a new term.

Definition 36 (C-compatible) *Let $\langle\mathcal{T},\mathcal{D}\rangle$ be a LHS-coherent defeasible KB, $R = \{\mathcal{D}_0,\ldots,\mathcal{D}_n\}$ its ranking and C a concept. Now, let $R' \subseteq R$ be a sequence of subsets $\mathcal{D}_n,\mathcal{D}_{n-1},\ldots,\mathcal{D}_i$ where $0 \leq i \leq n$. Then, R' (resp. $\mathcal{C}_{\bigcup R'}$) is* C-compatible *w.r.t. R if $\mathcal{T} \cup \mathcal{D}_n \cup \mathcal{D}_{n-1} \cup \ldots \cup \mathcal{D}_i \not\models_r \top \mathrel{\tilde{\sqsubset}} \neg C$. If there is no $j < i$ s.t. $\mathcal{T} \cup \mathcal{D}_n \cup \mathcal{D}_{n-1} \cup \ldots \cup \mathcal{D}_j \not\models_r \top \mathrel{\tilde{\sqsubset}} \neg C$ then we say R' (resp. $\mathcal{C}_{\bigcup R'}$) is* maximally *C-compatible w.r.t. R.*

Informally, Definition 36 on Page 172 describes the maximal amount of knowledge in the ranking (while respecting its ordering) that does not enforce exceptionality of a given concept. Notice that an immediate consequence of this, is the following lemma.

Lemma 10 (Rank and Maximal Compatibility Correspondence) *Given a defeasible KB $\langle\mathcal{T},\mathcal{D}\rangle$, its ranking $R = \{\mathcal{D}_0,\ldots,\mathcal{D}_n\}$ and a concept C. Then, $\texttt{Rank}(\langle\mathcal{T},\mathcal{D}\rangle, R, C) = i$ if and only if $\{\mathcal{D}_n,\ldots,\mathcal{D}_i\}$ is maximally C-compatible w.r.t. R for each $0 \leq i \leq n$.*

Given Definition 36 on Page 172 and Lemma 10 on Page 172, we can give an intuitive description of our alternative procedure for Rational Closure in the

following way: given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, its ranking R and a query axiom $C \mathrel{\sqsubseteq\!\!\!\sim} D$, to determine if $C \mathrel{\sqsubseteq\!\!\!\sim} D$ is in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$ all we have to do is (1) identify the maximal C-compatible subset of R and (2) look in all classical models for $\mathcal{T}$, if those objects that satisfy (the internalisation of) the C-compatible subset of R *and* C, also satisfy D. Then $C \mathrel{\sqsubseteq\!\!\!\sim} D$ is in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$, otherwise not. We have consolidated the mechanics of the algorithm described above in Procedure `RationalClosureB` on Page 173.

Procedure `RationalClosureB`$(\langle \mathcal{T}, \mathcal{D} \rangle, R, \delta)$

Input: A LHS-coherent defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, its ranking $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ and a defeasible subsumption $\delta = C \mathrel{\sqsubseteq\!\!\!\sim} D$.

Output: **true** if δ is in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$, **false** otherwise.

```
1 i := Rank(⟨T, D⟩, R, C);
2 if i = ∞ then
3     return T ⊨ C ⊑ D;
4 else
5     return T ⊨ C_{D_n ∪ ... ∪ D_i} ⊓ C ⊑ D;
```

Therefore, Procedure `RationalClosureB` on Page 173 works by progressively (or incrementally) ignoring facts in our KB, starting with the least exceptional (or least specific), until we reach a point where our knowledge allows us to consider the antecedent of our query axiom as prototypical. Thereafter, we check if the consequent is a property of this antecedent w.r.t. the remaining knowledge. Consider the student and employed student example:

Example 22 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$*:*

$$\left\langle \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\}, \left\{ \begin{array}{lll} \mathsf{Student} & \mathrel{\sqsubseteq\!\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} & \mathrel{\sqsubseteq\!\!\!\sim} & \exists \mathsf{receives.TaxInvoice} \end{array} \right\} \right\rangle$$

□

The ranking R for $\langle \mathcal{T}, \mathcal{D} \rangle$ in Example 22 on Page 173 is the sequence of subsets of $\mathcal{D}$:

$$\mathcal{D}_0 = \{\mathsf{Student} \quad \tilde{\sqsubseteq} \quad \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})\},$$
$$\mathcal{D}_1 = \{\mathsf{EmployedStudent} \quad \tilde{\sqsubseteq} \quad \exists \mathsf{receives}.\mathsf{TaxInvoice}\}$$

Suppose that we want to verify if axioms $\mathsf{Student} \tilde{\sqsubseteq} \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$, $\mathsf{EmployedStudent} \tilde{\sqsubseteq} \exists \mathsf{receives}.\mathsf{TaxInvoice}$ and $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \tilde{\sqsubseteq} \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$ are in the Rational Closure for $\langle \mathcal{T}, \mathcal{D} \rangle$. It is straightforward to determine, for each query antecedent C, the maximal C-compatible subsets of R.

$\{\mathcal{D}_0, \mathcal{D}_1, \mathcal{D}_2\}$ is maximally $\mathsf{Student}$-compatible,
$\{\mathcal{D}_1, \mathcal{D}_2\}$ is maximally $\mathsf{EmployedStudent}$-compatible,
$\{\mathcal{D}_1, \mathcal{D}_2\}$ is maximally $(\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top)$-compatible

It is easy to see that Line 1 of Procedure `RationalClosureB` on Page 173 is responsible for determining the maximal C-compatibility. Thereafter, we can execute classical DL reasoning to answer our queries (Line 5):

$\mathcal{T} \models \mathcal{C}_{\mathcal{D}_2 \cup \mathcal{D}_1 \cup \mathcal{D}_0} \sqcap \mathsf{Student} \sqsubseteq \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$,
$\mathcal{T} \models \mathcal{C}_{\mathcal{D}_2 \cup \mathcal{D}_1} \sqcap \mathsf{EmployedStudent} \sqsubseteq \exists \mathsf{receives}.\mathsf{TaxInvoice}$,
$\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_2 \cup \mathcal{D}_1} \sqcap \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \sqsubseteq \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$

Therefore we can conclude that the axioms $\mathsf{Student} \tilde{\sqsubseteq} \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$ and $\mathsf{EmployedStudent} \tilde{\sqsubseteq} \exists \mathsf{receives}.\mathsf{TaxInvoice}$ are in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$, while $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \tilde{\sqsubseteq} \neg(\exists \mathsf{receives}.\mathsf{TaxInvoice})$ is not.

Notice that Rational Closure reasoning (Procedure `RationalClosureA` and `RationalClosureB`), in the case where the antecedent concept of our query is *not* exceptional, looks at all objects in our models for $\mathcal{T}$ which satisfy all our defeasible subsumptions *and* the antecedent of our query. If the consequent of our query is satisfied by each of these objects then we can infer that the query axiom is in the Rational Closure of our KB[2].

[2]Recall from Section 4.1 that the internalisation mechanism is only necessary to allow the role neighbourhood of an object to violate defeasible subsumptions in $\mathcal{ALC}$. Because

We now show that Procedure `RationalClosureB` on Page 173 corresponds to Procedure `RationalClosureA` (they give back the same entailments). Note that showing this correspondence proves termination and soundness for Procedure `RationalClosureB`. We first prove the case where the antecedent of our query does not have infinite rank (according to both procedures) and then later address the infinite rank case. Lemma 11 on Page 175 shows that if $C \sqsubset\!\!\sim D$ is in the Rational Closure using Procedure `RationalClosureA` then it will be in the Rational Closure using Procedure `RationalClosureB`.

Lemma 11 (RationalClosureA vs. RationalClosureB, $\Longrightarrow$)
*Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, its ranking $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$, $C \sqsubset\!\!\sim D$ a query axiom and i, j natural numbers s.t. $i =$ **Rank**$(\langle \mathcal{T}, \mathcal{D} \rangle, R, C)$ and $j =$ **Rank**$(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D)$. If $i < j$ then $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C \sqsubseteq D$ for each $0 \leq i \leq n$.*

Proof: We have to show that if $i < j$ then $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C \sqsubseteq D$. Assume $i < j$ but $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C \sqsubseteq D$. We try to derive a contradiction. The latter statement implies that there is a classical model $\mathcal{I}$ for $\mathcal{T}$ s.t. there is an $x \in (\mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C)^{\mathcal{I}}$ s.t. $x \notin D^{\mathcal{I}}$. But, because we know that $i < j$ it implies that $C \sqcap \neg D$ would still be exceptional w.r.t. $\mathcal{T} \cup \mathcal{D}_n \cup \ldots \cup \mathcal{D}_i$ (it would only become non-exceptional w.r.t. $\mathcal{T} \cup \mathcal{D}_n \cup \ldots \cup \mathcal{D}_j$ where $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j \subseteq \mathcal{D}_n \cup \ldots \cup \mathcal{D}_i$). Therefore, $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqsubseteq \neg(C \sqcap \neg D)$. This is clearly in contradiction with our model $\mathcal{I}$ for $\mathcal{T}$. Therefore, $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C \sqsubseteq D$.$\square$

Now, we have to show that if $C \sqsubset\!\!\sim D$ is in the Rational Closure according to Procedure `RationalClosureB` on Page 173 then it will also be in the Rational Closure according to Procedure `RationalClosureA` on Page 171.

---

we don't have roles in propositional languages, Rational Closure is inferentially indistinguishable from classical reasoning in the case where the antecedent concept of our query is not exceptional w.r.t. our KB. This seems to be a very reasonable and desirable paradigm for many applications of defeasible reasoning.

Lemma 12 (RationalClosureA vs. RationalClosureB, $\Longleftarrow$)
Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $C \mathrel{\tilde{\sqsubseteq}} D$ a query axiom and i the lowest natural number s.t. $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqsubseteq \neg C$ (according to Procedure `RationalClosureB`*). Then, if $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C \sqsubseteq D$ then $\mathit{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) < \mathit{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D)$.*

Proof: We have to show that if $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C \sqsubseteq D$ then $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) < \mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D)$. Assume that $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C \sqsubseteq D$ but that $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) \not< \mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D)$. There are two cases:
Case 1: $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) = \mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D)$. We know that $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) = i$ because of Lemma 10 on Page 172. Because of our assumption that $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) = \mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D)$, we can infer that $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D) = i$. But this would imply that $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqsubseteq \neg(C \sqcap \neg D)$. This, in turn, means that there is a classical model $\mathcal{I}$ for $\mathcal{T}$ s.t. there is an $x \in (\mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i})^{\mathcal{I}}$ s.t. $x \in C^{\mathcal{I}}$ and $x \notin D^{\mathcal{I}}$. This is clearly in contradiction with our original assumption that $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C \sqsubseteq D$. Therefore, $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) \neq \mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D)$.
Case 2: $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) > \mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D)$. Notice that the term $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C)$ defines the lowest number i s.t. $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqsubseteq \neg C$. Hence $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D) = j$ where $j < i$. This implies that $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j} \sqsubseteq \neg(C \sqcap \neg D)$ where $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i \subset \mathcal{D}_n \cup \ldots \cup \mathcal{D}_j$. By definition of $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C)$, i is the *lowest* number s.t. C is not exceptional w.r.t. the ranking. Therefore, because $i > j$, C is still exceptional w.r.t. $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j$. In other words, $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j} \sqsubseteq \neg C$. This is clearly in contradiction with $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j} \sqsubseteq \neg(C \sqcap \neg D)$. Therefore, $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) \not> \mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C \sqcap \neg D)$.

We now turn our attention to the case where $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) = \infty$. We have to show, for any defeasible KB, that if $\mathtt{Rank}(\langle \mathcal{T}, \mathcal{D} \rangle, R, C) = \infty$ (according to Procedure `Rank` on Page 170), then $\mathcal{T} \models C \sqsubseteq D$ (Line 3 of Procedure `RationalClosureB`). This is straightforward to see from Lines $1-3$ of Procedure `RationalClosureB` on Page 173. The converse also holds by monotonicity of classical entailment. $\square$

In terms of the worst-case computational complexity of computing Rational Closure for $\mathcal{ALC}$ defeasible KBs, observe that we first need to compute the ranking of the KB (see Procedure `ComputeRankingB` on Page 157). Notice that, in the worst case, we do n exceptionality checks to determine $\mathcal{E}_1$, where n is the number of axioms in $\mathcal{E}_0$. In the worst case, only one axiom would not be exceptional in $\mathcal{E}_0$. If all of them are exceptional then we have a fixed point and we stop, therefore it cannot be the worst case. Similarly, if none of them are exceptional then our ranking only has one rank and we stop, so this cannot be the worst case either. Hence, to determine $\mathcal{E}_2$ we would have to do $n-1$ exceptionality checks and for $\mathcal{E}_3$, $n-2$ checks, and so on. The result is a quadratic number of exceptionality checks (classical $\mathcal{ALC}$ entailment checks) in the number of defeasible subsumptions in the KB.

After computing the ranking, one could use either of the aforementioned procedures for computing Rational Closure (Procedure `RationalClosureA` or `RationalClosureB`). It is clear that, in the worst case for these procedures, we have $n+1$ classical entailment checks where n is the number of ranks in the computed ranking of the KB. Thus we have a linear number of operations for these procedures. To compute the ranking itself requires a quadratic number of exceptionality checks in the size of $\mathcal{D}$ (if the KB is LHS-coherent). Notice that in the general case (when the KB is LHS-incoherent) we have to perform these quadratic number of checks each time we reach a fixed point with totally exceptional axioms. In the worst case, at each of these fixed points we will have one axiom from $\mathcal{D}$ moved to $\mathcal{T}$. Thus we will have to perform the quadratic number of exceptionality checks n times (where n is the size of $\mathcal{D}$). This gives a cubic complexity n^3 for the full ranking procedure. Therefore, in total we have a linear number of operations in the size of the ranking, with a cubic number of operations (in the number of defeasible subsumptions) to compute the ranking, on top of the decision procedure for classical $\mathcal{ALC}$ (which is in EXPTIME). The result is thus a procedure that still terminates in EXPTIME. Therefore, Ra-

tional Closure for $\mathcal{ALC}$ (with defeasible subsumption) is not in a higher complexity class than classical entailment for classical $\mathcal{ALC}$. One can also observe that Procedure `RationalClosureB` on Page 173 is likely to be less computationally intensive in practice, than Procedure `RationalClosureA` on Page 171. The reason is that Procedure `RationalClosureB` only requires to compute the rank for the antecedent concept of our query. Whereas, Procedure `RationalClosureA` requires to compute the ranks of *two* concepts.
To conclude this section we show that, although Rational Closure may be useful for a variety of applications, it sometimes cautious to draw inferences that may be useful in other applications. Consider the following example:

Example 23 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$ *where:*

$$\mathcal{T} = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \sqsubseteq & \mathsf{Student}, \\ \mathsf{SelfSponsoredStudent} & \sqsubseteq & \mathsf{Student} \end{array} \right\},$$

$$\mathcal{D} = \left\{ \begin{array}{lcl} \mathsf{Student} & \mathrel{\sim\!\!\sqsubseteq} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \mathrel{\sim\!\!\sqsubseteq} & \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{Student} & \mathrel{\sim\!\!\sqsubseteq} & \neg \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \mathrel{\sim\!\!\sqsubseteq} & \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \mathrel{\sim\!\!\sqsubseteq} & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \mathrel{\sim\!\!\sqsubseteq} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{worksFor.University} & \mathrel{\sim\!\!\sqsubseteq} & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\}$$

□

Applying the ranking procedure (Procedure `ComputeRankingB`) to the $\langle \mathcal{T}, \mathcal{D} \rangle$ in Example 23 on Page 178 we obtain the ranking:

$$\mathcal{D}_0 = \left\{ \begin{array}{lcl} \mathsf{Student} & \mathrel{\sim\!\!\sqsubseteq} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \mathrel{\sim\!\!\sqsubseteq} & \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{Student} & \mathrel{\sim\!\!\sqsubseteq} & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\},$$

$$\mathcal{D}_1 = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \mathrel{\sim\!\!\sqsubseteq} & \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \mathrel{\sim\!\!\sqsubseteq} & \exists \mathsf{receives.TaxInvoice} \end{array} \right\},$$

$$\mathcal{D}_2 = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \mathrel{\sim\!\!\sqsubseteq} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{worksFor.University} & \mathrel{\sim\!\!\sqsubseteq} & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\}$$

In Example 23 on Page 178 $\mathsf{EmployedStudent} \sqsubseteq \exists \mathsf{access.UniversityLibrary}$ is not in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$. However, there is a reasonable argument that a more adventurous (yet still sensible) reasoning paradigm *should* endorse this conclusion. The issue is that Rational Closure defines a reasoning paradigm which roughly corresponds to: "I will only conclude something if I have conclusive evidence to do so". In other words, if something is exceptional according to Rational Closure, Rational Closure will only allow it to inherit properties enforced by axioms of the same degree of exceptionality. Hence, employed students are not allowed to inherit the property of having access to a university library from general students.

Similar behaviour dictates that employed students who have children (as well as employed students who work for a university) do not have access to a university library, when it seems reasonable (using adventurous reasoning), to conclude that they do. Analogously, adventurous reasoning should be able to conclude that employed students that have children usually are self sponsored, because this is a typical property of employed students. By the same token, our knowledge does not preclude us from inferring that employed students who work for a university are obliged to pay taxes.

Our arguments boil down to the fact that there are no constraints which prevent one from making the above inferences, therefore, a more adventurous reasoning methodology could permit them. For example, just because employed students are exceptional, an adventurous reasoning paradigm may argue that the *reason* why they are exceptional is related to their tax paying property (irrelevant to the property of having access to a library), and therefore one may conclude that they have access to a library.

Example 23 on Page 178 demonstrates that Rational Closure defines a more skeptical or cautious reasoning paradigm in the presence of exceptions. This does not mean that Rational Closure is not useful. Rather, it is recommended for applications where this reasoning paradigm is more suitable. We stated a case for more adventurous reasoning that may be used to conclude more inferences than those that are endorsed by Rational Closure. Such a

paradigm is motivated by the fact that there are no constraints which *prevent* the endorsement of these conclusions. In other words, one may be interested in a reasoning paradigm which is roughly closer to: "I will assume that something can be inferred, unless there is explicit knowledge to the contrary".

This latter paradigm is one that lies on the opposite side of the spectrum to Rational Closure. It can be seen as a venturous extension to Rational Closure's skeptical entailment regime. In the next section we present a rational consequence relation which defines such a reasoning paradigm. It is an adaptation to DLs of the presumptive reasoning paradigm, developed by Daniel Lehmann [116], called the *Lexicographic Closure*.

4.4 Lexicographic Closure

The Lexicographic Closure (LC) of a defeasible KB defines an entailment regime that represents a venturous extension to Rational Closure's skeptical inference. That is, LC will give back all the positive inferences that Rational Closure gives but it will also give back additional inferences. Nevertheless, LC remains a rational consequence relation satisfying all the KLM postulates. As we will see later, LC is a syntax-dependent construction.

More specifically, if the goal is to derive the maximal number of positive inferences, LC favours finer granularity of axioms in the KB construction. That is, given two defeasible KBs $\mathcal{K}_1$ and $\mathcal{K}_2$ which are logically equivalent (having the same ranked models), supposing that the granularity of knowledge in $\mathcal{K}_2$ is finer than that of $\mathcal{K}_1$, then it is possible that LC will give more positive inferences for $\mathcal{K}_2$ than for $\mathcal{K}_1$. We demonstrate this property with examples at the end of this section.

We first provide a semantics for Lexicographic Closure which is actually an incremental adaptation of the semantics of Rational Closure based on its canonical model construction. This semantic characterisation is also a generalisation of the one given by Daniel Lehmann for the propositional case [116]. Thereafter, we provide an algorithmic construction of LC based

on the one by Casini and Straccia [52] and prove the correspondence with its semantics.

4.4.1 Semantics

Given an $\mathcal{ALC}$ defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, we will define a minimal canonical model for $\langle \mathcal{T}, \mathcal{D} \rangle$ and propose this model as the base for characterising a semantics for Lexicographic Closure. We saw that Rational Closure can be characterised by a single minimal canonical model for $\langle \mathcal{T}, \mathcal{D} \rangle$ (see Definition 23 on Page 97). For brevity, we shall refer to this model as the *Rational Closure defining model* (RCDM for short) for a defeasible KB. Intuitively, we propose a refinement of this model in which we modify only the ordering on the elements in the domain.

Recall from the end of Section 4.2, that in order to guarantee the computation of a rational consequence relation, the exceptionality ranking of the KB should be respected. Nevertheless, we demonstrated that it was still possible to refine the ranking by distinguishing between sentences of the same rank. Of course, this description is from the algorithmic perspective. From the model-theoretic perspective, such a refinement can be perceived in a ranked model for the KB, as distinguishing between objects of the domain that have the same rank.

The semantics we propose for Lexicographic Closure adopts this view. Though the question still remains: what criteria do we use to distinguish between objects of the same rank? In principle, there is a multitude of criteria to choose from. However, for Lexicographic Closure we choose to focus on the *number of sentences* that are *classically* satisfied by an object in a ranked model.

As we saw with Rational Closure, algorithmically speaking, the nonmonotonic nature of the reasoning mechanism is realised through ignorance of knowledge that is inconsistent with the query being asked. Intuitively, using the maximal amount of knowledge at our disposal should, in general, lead to more accurate inferences. Therefore, the idea is to ignore the least amount of

knowledge as possible. The intuition behind counting the sentences that are classically satisfied by an object in a ranked model, is to arrive at a refined ranking of objects in the domain taking into account this factor. This idea, in turn, hopefully leads to a semantics that could allow us to retain more sentences algorithmically. Consider the following example.

Example 24 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$*:*

$$\left\langle \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\}, \left\{ \begin{array}{lcl} \mathsf{Student} & \mathrel{\tilde{\sqsubseteq}} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{EmployedStudent} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{receives.TaxInvoice} \end{array} \right\} \right\rangle$$

□

For brevity and readability, we use symbols S, E, U and T to refer to the concept names Student, EmployedStudent, UniversityLibrary and TaxInvoice respectively in Example 24 on Page 182. Similarly, we use the abbreviations *acc.* and *rec.* to refer to the role names access and receives respectively. Consider the following ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$.

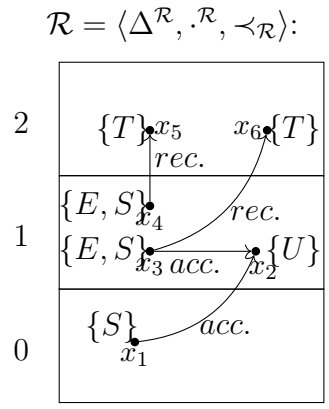


Figure 4.3: Non-refined ranked model for KB of Example 24.

We can encode the model in Figure 4.3 as $\mathcal{R} = \langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}}, \prec_{\mathcal{R}} \rangle$ where $\Delta^{\mathcal{R}} = \{x_1, x_2, x_3, x_4, x_5, x_6\}$, $S^{\mathcal{R}} = \{x_1, x_3, x_4\}$, $E^{\mathcal{R}} = \{x_3, x_4\}$, $U^{\mathcal{R}} = \{x_2\}$, $T^{\mathcal{R}} = \{x_5, x_6\}$, $acc.^{\mathcal{R}} = \{(x_1, x_2), (x_3, x_2)\}$ and $rec.^{\mathcal{R}} = \{(x_3, x_6), (x_4, x_5)\}$. $rk_{\mathcal{R}}(x_1) = 0$, $rk_{\mathcal{R}}(x_2) = rk_{\mathcal{R}}(x_3) = rk_{\mathcal{R}}(x_4) = 1$ and $rk_{\mathcal{R}}(x_5) = rk_{\mathcal{R}}(x_6) = 2$.

Given this representation, we can observe that $\mathcal{R} \not\Vdash E \sqsubseteq \!\!\!\sim \exists acc.U$ even though it makes intuitive sense to be able to infer $E \sqsubseteq \!\!\!\sim \exists acc.U$ from our KB. In other words, there is no information in my KB that prevents me from inferring that employed students have access to a university library. The reason why $\mathcal{R}$ does not satisfy this axiom is that x_4 violates it. In other words $x_4 \notin (\neg E \sqcup \exists acc.U)^{\mathcal{R}}$ (x_4 is not in the materialised concept representing the axiom $E \sqsubseteq \!\!\!\sim \exists acc.U$). Observe that if we look at the materialised concept for each defeasible subsumption in the KB, we can determine the *number* of these materialised concepts that each object in $\Delta^{\mathcal{R}}$ satisfies. These numbers are $3, 3, 2, 1, 3, 3$ for $x_1, x_2, x_3, x_4, x_5, x_6$ respectively. Now, even though x_3 and x_4 (and x_2) have the same rank, it is possible to consider x_3 more "normal" in a sense than x_4 because x_3 satisfies more materialised concepts from the KB than x_4.

More precisely, we can refine the second rank (rank 1) of $\mathcal{R}$ to include three sub-ranks. The most typical of these sub-ranks would include element x_2 since it satisfies the most materialised concepts (i.e., three). The second, higher sub-rank would include element x_3 because it satisfies two materialised concepts. Finally, x_4 would be present in the highest sub-rank because it satisfies just one materialised concept. This refined ranked model $\mathcal{R}'$ is depicted below. Note that we do not need to consider the subsumptions in $\mathcal{T}$ because *every* object in a ranked model has to satisfy the applicable materialised concepts for $\mathcal{T}$, while this is not necessarily the case for the defeasible subsumptions, as we have demonstrated.
Considering $\mathcal{R}'$ in Figure 4.4 it is easy to verify that $\mathcal{R}' \Vdash E \sqsubseteq \!\!\!\sim \exists acc.U$, which captures our earlier intuition about employed students having access to a university library.

Essentially, Example 24 on Page 182 presents the basic idea of the semantic paradigm we want to capture with Lexicographic Closure. What remains is to formalise the construction in Example 24 and give a concrete semantic definition for Lexicographic Closure. The first step is to define the *lexicographic ordering* on the objects in a ranked model for a defeasible KB.

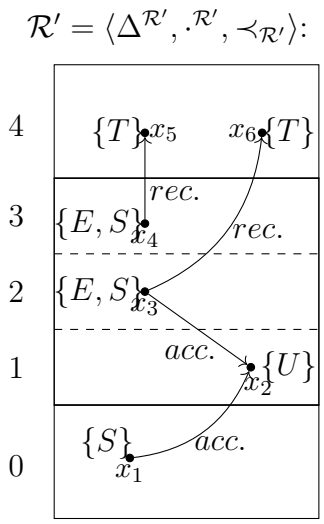


Figure 4.4: Refined ranked model for KB of Example 24.

Definition 37 (Cardinality Ordering) *We let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $\mathcal{R}$ a ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$ and $x \in \Delta^{\mathcal{R}}$. The* cardinality rank *of x in $\mathcal{R}$ w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$, denoted by $lrk_{\mathcal{R}}(x)$, is the natural number defined by the cardinality of the following set $\{C \sqsubset D \in \mathcal{D} \mid x \in (\neg C \sqcup D)^{\mathcal{R}}\}$. The* cardinality ordering *for $\mathcal{R}$ w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$, denoted by $\prec_{lexi_{\mathcal{R}}}$, is a total pre-order on the elements of $\Delta^{\mathcal{R}}$ defined by the function lrk w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$. That is, for any two elements x and y in $\Delta^{\mathcal{R}}$, $x \prec_{lexi_{\mathcal{R}}} y$ if and only if $lrk_{\mathcal{R}}(x) < lrk_{\mathcal{R}}(y)$.*

Informally, Definition 37 on Page 183 assigns a natural number to each element in the domain that represents the number of defeasible subsumptions that it satisfies in the KB $\langle \mathcal{T}, \mathcal{D} \rangle$. As we demonstrated in Example 24 on Page 182, the idea is to use the ordering defined by a standard ranked model as a base and, for the objects which have the same rank, we can refine their ranks by taking the *lexicographic ordering* over these two orderings (see Definition 38 on Page 185).

Since it was shown that, for $\mathcal{ALC}$, Rational Closure can be characterised by a single canonical ranked model for any consistent KB $\langle \mathcal{T}, \mathcal{D} \rangle$, we can build the semantics of Lexicographic Closure upon this model. We will now

define a ranked model that is a "lexicographic refinement" of this model. This model, which we call the *Lexicographic Closure defining model* (LCDM) for $\langle \mathcal{T}, \mathcal{D} \rangle$, will be the base-point for defining Lexicographic Closure.

Definition 38 (Lexicographic Closure Defining Model) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $\mathcal{R} = \langle \Delta^{\mathcal{R}}, \cdot^{\mathcal{R}}, \prec_{\mathcal{R}} \rangle$ the RCDM for $\langle \mathcal{T}, \mathcal{D} \rangle$. The* Lexicographic Closure defining model *(LCDM) for $\langle \mathcal{T}, \mathcal{D} \rangle$ is defined as $\mathcal{R}' = \langle \Delta^{\mathcal{R}'}, \cdot^{\mathcal{R}'}, \prec_{\mathcal{R}'} \rangle$ where $\Delta^{\mathcal{R}'} = \Delta^{\mathcal{R}}$, $\cdot^{\mathcal{R}'} = \cdot^{\mathcal{R}}$ and $\prec_{\mathcal{R}'}$ is defined on $\Delta^{\mathcal{R}'}$ as follows: for each $(a, b) \in \Delta^{\mathcal{R}'} \times \Delta^{\mathcal{R}'}$, $a \prec_{\mathcal{R}'} b$ if $a \prec_{\mathcal{R}} b$ or $a \prec_{lexi_{\mathcal{R}}} b$.*

It is easy to see that, given that the RCDM exists for some $\langle \mathcal{T}, \mathcal{D} \rangle$, the LCDM (Definition 38 on Page 185) also exists for $\langle \mathcal{T}, \mathcal{D} \rangle$. From the definition of LCDM it is straightforward to show that Lexicographic Closure is an inferential extension of Rational Closure. That is, we will now show that Lexicographic Closure gives back all the positive inferences that Rational Closure gives (and possibly others).

Lemma 13 (Lexicographic Closure Extends Rational Closure) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $\mathcal{R}$ the RCDM for $\langle \mathcal{T}, \mathcal{D} \rangle$ and $\mathcal{R}'$ the LCDM $\langle \mathcal{T}, \mathcal{D} \rangle$. Then, if $\mathcal{R} \Vdash C \sqsubseteq D$ then $\mathcal{R}' \Vdash C \sqsubseteq D$ for any $C \sqsubseteq D$.*

Proof: Assume that $\mathcal{R} \Vdash C \sqsubseteq D$. Of course we know that $\mathcal{R}$ and $\mathcal{R}'$ only differ in the ordering on the domain elements (that is, $\Delta^{\mathcal{R}} = \Delta^{\mathcal{R}'}$ and $\cdot^{\mathcal{R}} = \cdot^{\mathcal{R}'}$). It is also apparent from the definition of lexicographic refinement on the ordering that $\min_{\prec_{\mathcal{R}'}}(C^{\mathcal{R}'}) \subseteq \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}})$. Therefore $\mathcal{R}' \Vdash C \sqsubseteq D$. □

Given the notion of LCDM, we can define Lexicographic Closure in an analogous way to Rational Closure (Definition 20 on Page 95).

Definition 39 (Lexicographic Closure of a Defeasible KB) *Given a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, its LCDM $\mathcal{R}$ and a defeasible subsumption $C \sqsubseteq D$, $C \sqsubseteq D$ is in the* Lexicographic Closure *of $\langle \mathcal{T}, \mathcal{D} \rangle$ (denoted by $\langle \mathcal{T}, \mathcal{D} \rangle \models_{lexico} C \sqsubseteq D$) if $rk_{\mathcal{R}}(C) < rk_{\mathcal{R}}(C \sqcap \neg D)$ or $rk_{\mathcal{R}}(C) = \infty$.*

Thus we have given a semantics for Lexicographic Closure by adapting the intuitions of Daniel Lehmann [116] in the propositional setting, to the DL case. In the next section we give a procedure for computing Lexicographic Closure and prove its correspondence with the semantics proposed in this section.

4.4.2 Procedure

In the DL case, there has been very little work on defining Lexicographic Closure (in the sense of Daniel Lehmann [116]). The only known application to DLs is the procedure developed by Casini and Straccia [52]. In this section we devise a variant of this procedure which aligns with the semantic foundation of preferential DLs (as delineated in Section 2.8) as well as with the semantics for Lexicographic Closure proposed in the previous section.

The lexicographic refinement of the ordering on objects of a ranked model (Definition 37 on Page 183) manifests itself, on the level of sentences in the KB, as a refinement of the ordering on these sentences (the exceptionality ordering). Consider the following example.

Example 25 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$*:*

$$\left\langle \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\}, \left\{ \begin{array}{lcl} \mathsf{Student} & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \precsim & \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{EmployedStudent} & \precsim & \exists \mathsf{receives.TaxInvoice} \end{array} \right\} \right\rangle$$

□

The KB in Example 25 on Page 186 is the same as the one in Example 24 on Page 182. In the latter example we showed that the ordering on the objects in a ranked model can be refined to reflect the number of sentences in the KB that each object satisfies (classically). On the level of sentences, and specifically with regards to the ranking of a defeasible KB, we advocate that sentences with the same rank can be distinguished using the tool of lexicographic ordering mentioned in Section 4.4.1. It is easy to see that the ranking for $\langle \mathcal{T}, \mathcal{D} \rangle$ is:

$$\mathcal{D}_0 = \left\{ \begin{array}{lcl} \mathsf{Student} & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \precsim & \exists \mathsf{access.UniversityLibrary}, \end{array} \right\},$$

$$\mathcal{D}_1 = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \precsim & \exists \mathsf{receives.TaxInvoice} \end{array} \right\}$$

Now, imagine that our query is concerning employed students. That is, our query is of the form $\mathsf{EmployedStudent} \precsim X$ for some concept X. It is straightforward to see that, in order to find the maximal EmployedStudent-compatible information in the ranking, Rational Closure will eliminate *all* information in $\mathcal{D}_0$. Intuitively, some might argue that this is too "drastic". That is, even though it is reasonable to eliminate $\mathsf{Student} \precsim \neg(\exists \mathsf{receives.TaxInvoice})$ (because it is contributing to the exceptionality of the concept EmployedStudent), it is not as reasonable to eliminate $\mathsf{Student} \precsim \exists \mathsf{access.UniversityLibrary}$. This is because the latter information has nothing to do with the exceptionality of EmployedStudent.

Therefore, one can view this elimination behaviour of Rational Closure as quite "coarse". The idea behind Lexicographic Closure, from an algorithmic perspective, is to take a "fine-grained" approach to this elimination of information. Recall that the goal is to determine the maximally C-compatible subset of the ranking (for a concept C). Therefore, instead of removing "whole ranks" at a time to arrive at this set, Lexicographic Closure starts by removing a single axiom from the lowest rank (there are k ways to do this if k is the number of axioms in the particular rank). If this is not sufficient to ensure C-compatibility we try to remove two axioms from the rank (to borrow a phrase from combinatorics, there are "k choose 2" ways to do this). This procedure continues until we have C-compatibility. If we have eliminated all k axioms from the rank then we move to the next rank and so on. It is not difficult to see that we have a combinatorial explosion of operations in the size of the ranks.

Applied to our example KB, we notice that there are only two axioms in $\mathcal{D}_0$ and so we have a simple case. We have to try removing a single axiom from

$\mathcal{D}_0$. There are two ways to do this because there are only two axioms. Removing $\mathsf{Student} \sqsubseteq \exists \mathsf{access.UniversityLibrary}$ does not result in $\mathsf{EmployedStudent}$-compatibility, however, removing $\mathsf{Student} \sqsubseteq \neg(\exists \mathsf{receives.TaxInvoice})$ does. Therefore we only need to remove the latter and we are left with:

$$\mathcal{D}_0 = \Big\{ \mathsf{Student} \quad \sqsubseteq \quad \exists \mathsf{access.UniversityLibrary} \Big\},$$

$$\mathcal{D}_1 = \Big\{ \mathsf{EmployedStudent} \quad \sqsubseteq \quad \exists \mathsf{receives.TaxInvoice} \Big\}$$

From this ranking we can infer $\mathsf{EmployedStudent} \sqsubseteq \exists \mathsf{access.UniversityLibrary}$ follows from $\mathcal{T} \cup \mathcal{D}_0 \cup \mathcal{D}_1$. Although this is simplistic example, we can easily extend this notion for the case where we have k axioms in the rank. However, before we demonstrate the general case, we will state what it means to take the conjunction (which we denote by $\wedge$), or disjunction (which we denote by $\vee$), of a set of DL axioms. In actuality, when refer to the conjunction (resp. disjunction) of axioms, we are in fact referring to the conjunction (resp. disjunction) of their *materialised counterpart concepts* (see Definition 29 on Page 139). That is, the conjunction of the axioms $C_1 \sqsubseteq D_1$ and $C_2 \sqsubseteq D_2$ is actually the conjunction of their materialised counterpart concepts $\neg C_1 \sqcup D_1$ and $\neg C_2 \sqcup D_2$ respectively. In other words, their conjunction is the concept $(\neg C_1 \sqcup D_1) \sqcap (\neg C_2 \sqcup D_2)$. Similarly, the disjunction of these axioms is the concept $(\neg C_1 \sqcup D_1) \sqcup (\neg C_2 \sqcup D_2)$.

Consider the case of three axioms: i.e., let $\mathcal{D}'_0$ be the first rank in some ranking for a defeasible KB. And assume that $\mathcal{D}'_0$ contains three axioms α_1, α_2 and α_3. Logically speaking, Lexicographic Closure will first check if $\alpha_1 \wedge \alpha_2 \wedge \alpha_3$ (the combination of all three constraints, together with the rest of the knowledge in the ranking) enforces exceptionality of the query. If it does, then it will check if $(\alpha_1 \wedge \alpha_2) \vee (\alpha_1 \wedge \alpha_2) \vee (\alpha_2 \wedge \alpha_3)$ enforces exceptionality (all combinations of two of the three constraints). If it does, then it will check if $\alpha_1 \vee \alpha_2 \vee \alpha_3$ enforces exceptionality (all combinations of one constraint). If it does, then it will eliminate α_1, α_2 and α_3 and proceed to $\mathcal{D}'_1$ and repeat the process.

Observe that the Lexicographic Closure procedure follows the same basic iteration as the Rational Closure procedure. The main difference is that, during the phase of identifying C-compatibility, the procedure replaces the axioms in a rank (i.e., the "conjunction of the axioms" in the set) with progressively "weakened" versions of this information until we reach C-compatibility.

The discussion following Example 25 on Page 186 gives a general intuition behind the Lexicographic Closure algorithm that we are going to propose. However, we now need to formalise this procedure. The first step is to define the "lexicographic refinement" of an exceptionality ranking for a defeasible KB. Recall from Example 25 that we need to enumerate all ways of removing $1 \leq k \leq n$ axioms from a rank (set of axioms of size n) of the exceptionality ranking. We accomplish this by referring to the *powerset* of the rank, and to the cardinalities of the sets appearing in this powerset. We can thus define the *Lexicographicalisation* of an exceptionality ranking for use in the Lexicographic Closure procedure.

Definition 40 (Lexicographicalisation of a Ranking) *Consider a LHS-coherent defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, let $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ be its ranking, let $\mathcal{P}(X)$ (resp. $|X|$) denote the powerset (resp. cardinality) of an arbitrary set of elements X. Then, for $0 \leq i \leq n$, the* Lexicographicalisation *of $\mathcal{D}_i$ is a sequence of collections $\mathscr{D}'_i = \{\mathcal{D}'_1, \ldots, \mathcal{D}'_m\}$ ($m = |\mathcal{D}_i|$) s.t. for $1 \leq k \leq m$, $\mathcal{D}'_k = \{\mathcal{S} \in \mathcal{P}(\mathcal{D}_i) \mid |\mathcal{S}| = k\}$. $R' = \{\mathscr{D}'_0, \ldots, \mathscr{D}'_n\}$ is called the* Lexicographicalisation *of the ranking R, where $\mathscr{D}'_i$ is the Lexicographicalisation of $\mathcal{D}_i \in R$ for $0 \leq i \leq n$.*

Given a standard rank $\mathcal{D}_i$ of size n, Definition 40 on Page 189 allows us to refer to all subsets of $\mathcal{D}_i$ of size $1 \leq k \leq n$. In effect, this allows us to enumerate all ways of removing k axioms from the rank, which we can use in our procedure for computing Lexicographic Closure. For example, given a rank $\mathcal{D}_i = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\}$ where each α_i is an axiom, we can calculate the Lexicographicalisation $\mathscr{D}'_i$ for $\mathcal{D}_i$ as the sequence of collections:

$$\mathcal{D}'_1 = \{\{\alpha_1\}, \{\alpha_2\}, \{\alpha_3\}, \{\alpha_4\}, \{\alpha_5\}\},$$
$$\mathcal{D}'_2 = \{\{\alpha_1, \alpha_2\}, \{\alpha_1, \alpha_3\}, \{\alpha_1, \alpha_4\}, \{\alpha_1, \alpha_5\}, \{\alpha_2, \alpha_3\},$$
$$\{\alpha_2, \alpha_4\}, \{\alpha_2, \alpha_5\}, \{\alpha_3, \alpha_4\}, \{\alpha_3, \alpha_5\}, \{\alpha_4, \alpha_5\}\},$$
$$\mathcal{D}'_3 = \{\{\alpha_1, \alpha_2, \alpha_3\}, \{\alpha_1, \alpha_2, \alpha_4\}, \{\alpha_1, \alpha_2, \alpha_5\}, \{\alpha_1, \alpha_3, \alpha_4\}, \{\alpha_1, \alpha_3, \alpha_5\},$$
$$\{\alpha_1, \alpha_4, \alpha_5\}, \{\alpha_2, \alpha_3, \alpha_4\}, \{\alpha_2, \alpha_3, \alpha_5\}, \{\alpha_2, \alpha_4, \alpha_5\}, \{\alpha_3, \alpha_4, \alpha_5\}\},$$
$$\mathcal{D}'_4 = \{\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}, \{\alpha_1, \alpha_2, \alpha_3, \alpha_5\}, \{\alpha_1, \alpha_2, \alpha_4, \alpha_5\}, \{\alpha_1, \alpha_3, \alpha_4, \alpha_5\}, \{\alpha_2, \alpha_3, \alpha_4, \alpha_5\}\},$$
$$\mathcal{D}'_5 = \{\{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\}\}$$

In particular, notice that for any $\mathcal{D}_i$ and its Lexicographicalisation $\mathscr{D}'_i = \{\mathcal{D}'_1, \ldots, \mathcal{D}'_m\}$ where $m = |\mathcal{D}_i|$, $\mathcal{D}'_1$ will be a collection of singleton sets (each set corresponds to an axiom from $\mathcal{D}_i$), and $\mathcal{D}'_m$ will be a singleton collection consisting of the set $\mathcal{D}_i$ and only $\mathcal{D}_i$. Again, we reiterate that $\mathcal{D}'_k$ (for $1 \leq k \leq m$) represents an enumeration of "all ways of removing k axioms from a set of m axioms". Depending on how we use these sets, the dual of this statement could apply (i.e., it could also mean "all ways of *retaining* (or preserving) k axioms from a set of m axioms").

Now, we mention an important aspect of the lexicographic procedure which leads to a simple optimisation. We notice that we do not need to consider the Lexicographicalisation of *each* rank in the ranking. That is, Rational Closure is too coarse in determining maximal C-compatibility (using Procedure `Rank`). In other words, in the last step of this procedure, when we are eliminating the last rank from the ranking that is contributing to the exceptionality of C, there are possibly some irrelevant axioms in this last rank which we should keep. I.e., this is where the notion of Lexicographicalisation comes in. To take a fine-grained look at this *problematic rank* of the ranking.

Definition 41 (Problematic Rank) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its ranking, $C \sqsubseteq D$ a query and $\{\mathcal{D}_i, \ldots, \mathcal{D}_n\}$ the maximal C-compatible subset of R w.r.t. $C \sqsubseteq D$ (w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$), where $0 \leq i \leq n$. If $i > 0$ then the problematic rank w.r.t. $C \sqsubseteq D$ (w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$) is the set $\mathcal{D}_{i-1}$. If $i = 0$ then the problematic rank is the empty set.*

In other words, the problematic rank is the last rank thrown out by the procedure which computes maximal C-compatibility. Given that we have formalised the main differences of the Lexicographic Closure procedure (w.r.t. the

Rational Closure procedure) we are in a position to present a procedure for computing the construction. However, before we do so, we introduce some foundational definitions.

Recall that Rational Closure coarsely eliminates "whole ranks" in order to arrive at C-compatibility for the query. Given a standard ranking $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ for some KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and a query $C \sqsubseteq D$, suppose that $R' = \{\mathcal{D}_n, \ldots, \mathcal{D}_2\}$ is maximally C-compatible for $\langle \mathcal{T}, \mathcal{D} \rangle$. This means that $\mathcal{D}_1$ is the problematic rank for $\langle \mathcal{T}, \mathcal{D} \rangle$ which, in turn, means that $\{\mathcal{D}_n, \ldots, \mathcal{D}_2, \mathcal{D}_1\}$ is not C-compatible for $\langle \mathcal{T}, \mathcal{D} \rangle$. Therefore, according to Lexicographic Closure, it may be too strong to eliminate the whole of the problematic rank $\mathcal{D}_1$. That is, perhaps it is just *some* of the axioms in $\mathcal{D}_1$ that are problematic and need to be eliminated. In order to have a finer-grained view of $\mathcal{D}_1$ we need to look at its Lexicographicalisation (Definition 40 on Page 189). In particular, the goal is to locate the maximal amount of information from $\mathcal{D}_1$ that we can retain. This information is defined by the *lexicographically additive concept* (LAC) for the problematic rank w.r.t. R' and C.

Definition 42 (Lexicographically Additive Concept) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $C \sqsubseteq D$ a query, $R' = \{\mathcal{D}_n, \ldots, \mathcal{D}_i\}$ the maximal C-compatible subset of the ranking $\{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$, and $\mathscr{D}'_{i-1}\{\mathcal{D}'_1, \ldots, \mathcal{D}'_m\}$ the Lexicographicalisation of the problematic rank $\mathcal{D}_{i-1}$. Let k be the largest number s.t. $1 \leq k \leq m-1$ and $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap (\bigsqcup \mathcal{C}_{\mathcal{S} \in \mathcal{D}'_k}) \sqsubseteq \neg C$. Then, the concept $(\bigsqcup \mathcal{C}_{\mathcal{S} \in \mathcal{D}'_k})$ is known as the* lexicographically additive concept *(LAC) for $\mathcal{D}_{i-1}$ w.r.t. R' and C. $\mathcal{D}'_k$ is called the* lexicographically additive subset *(LAS) of $\mathcal{D}_{i-1}$ w.r.t. R' and C.*

The LAC simply represents the maximal knowledge from the problematic rank which is still compatible with the query. Notice that $1 \leq k \leq m-1$ and not $1 \leq k \leq m$. This is because if $k = m$ then it implies that we are keeping all m axioms from the problematic rank. It is clear that this shouldn't be allowed because the maximal C-compatible subset is $\{\mathcal{D}_n, \ldots, \mathcal{D}_i\}$ and

therefore $\{\mathcal{D}_n, \ldots, \mathcal{D}_i, \mathcal{D}_{i-1}\}$ will not be C-compatible. Given the definition of LAC, that maximal C-compatibility is defined in terms of exceptionality, and that exceptionality is reducible to classical DL entailment (Theorem 1 on Page 143), we can formulate a description of Lexicographic Closure in terms of classical DL entailment.

Definition 43 (Lexicographic Closure w.r.t. Classical Entailment) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $C \sqsubseteq\!\!\!\!\sim D$ a query, $R' = \{\mathcal{D}_n, \ldots, \mathcal{D}_i\}$ the maximal C-compatible subset of its ranking $\{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$, and $C_{R'}$ the LAC for $\mathcal{D}_{i-1}$ w.r.t. R' and C. Then, $C \sqsubseteq\!\!\!\!\sim D$ is in the Lexicographic Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$ if $\mathcal{T} \models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$.*

Now we can give the pseudocode algorithm for computing Lexicographic Closure. The algorithm is composed of Procedure `LexicographicClosureA` and a sub-procedure `LAC`. The latter sub-procedure is responsible for computing the LAC for the given antecedent of the query, while the former consolidates all relevant components of the procedure (as described in Definition 43 on Page 192).

We reiterate that $\{\mathcal{D}_n, \ldots, \mathcal{D}_i, \mathcal{D}_{i-1}\}$ is not C-compatible and so we cannot keep all axioms from $\mathcal{D}_{i-1}$, we have to remove at least one. Therefore we assign $k = m - 1$ (keep $m - 1$ of the m axioms in $\mathcal{D}_{i-1}$) on Line 1 of Procedure `LAC`, instead of assigning $k = m$. If k reaches 0 (Line 4 of Procedure `LAC`) then it corresponds to the case of not keeping any axioms from $\mathcal{D}_{i-1}$ which means the LAC should, theoretically speaking, be the empty concept $\bot$. However, as we shall see later, we use the LAC in the Lexicographic Closure procedure by taking its *conjunction* with the maximal C-compatible *concept* for the ranking. Of course the conjunction of any concept with the $\bot$ concept will return $\bot$ and this is not the behaviour we want to capture. Rather, we would like Lexicographic Closure to revert to using only the standard maximal C-compatible concept for the ranking, if it cannot find any information to keep from $\mathcal{D}_{i-1}$. Therefore, we assign the $\top$ concept to the LAC for this case. In other words, the conjunction of $\top$

Procedure `LAC`$(\langle \mathcal{T}, \mathcal{D} \rangle, \delta, R', \mathscr{D}'_{i-1})$

Input: A LHS-coherent defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, a query $\delta = C \sqsubseteq D$, the maximal C-compatible subset $R' = \{\mathcal{D}_n, \ldots, \mathcal{D}_i\}$ of the ranking $\{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ $(1 \leq i \leq n-1)$, and the Lexicographicalisation $\mathscr{D}'_{i-1} = \{\mathcal{D}'_1, \ldots, \mathcal{D}'_m\}$ for the problematic rank $\mathcal{D}_{i-1}$.

Output: LAC for $\mathcal{D}_{i-1}$ w.r.t. R' and C.

```
1 k := m - 1;
2 while T ⊨ C_⋃R' ⊓ (⨆ C_S∈D'_k) ⊑ ¬C do
3     k := k - 1;
4     if k = 0 then
5         return ⊤;
6 return (⨆ C_S∈D'_k);
```

with the maximal C-compatible concept for the ranking will return the latter concept. Given this sub-procedure for computing the LAC, we can devise a non-naïve algorithm (although still having substantial headroom for optimisation) for computing Lexicographic Closure. The pseudocode is represented in Procedure `LexicographicClosureA`.

Procedure `LexicographicClosureA` works by first finding the standard rank of the antecedent of our query (and corresponding maximal C-compatible subset of the ranking). This is done in Line 1 using Procedure `Rank`. Notice that this behaviour is analogous to the Rational Closure procedure. The departure point is Lines $2-3$ which take care of some special cases. The first is the case where $i = 0$. This means that the antecedent concept is not exceptional and therefore we check entailment w.r.t. all the knowledge (we do not need to eliminate any axioms). The same can be said of the case where the problematic rank just has a single axiom. In the latter case, there is only one way to repair the exceptionality of C, and that is to remove the single axiom. This leaves us with the standard maximal C-compatible concept.

Procedure `LexicographicClosureA`$(\langle \mathcal{T}, \mathcal{D} \rangle, R, R', \delta)$

Input: A LHS-coherent defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, its ranking $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$, a query $\delta = C \mathrel{\sqsubset\!\!\!\sim} D$, the problematic rank $\mathcal{D}_{i-1}$ and its Lexicographicalisation $\mathscr{D}'_{i-1} = \{\mathcal{D}'_1, \ldots, \mathcal{D}'_m\}$.

Output: **true** if δ is in the Lexicographic Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$, **false** otherwise.

1 $i :=$ `Rank`$(\langle \mathcal{T}, \mathcal{D} \rangle,\ R,\ C)$;
2 **if** $i = 0$ **or** $|\mathcal{D}_{i-1}| = 1$ **then**
3 $\quad$ **return** $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C \sqsubseteq D$;
4 **else**
5 $\quad C_{R'} =$ `LAC`$(\langle \mathcal{T}, \mathcal{D} \rangle, \delta, \{\mathcal{D}_n, \ldots, \mathcal{D}_i\}, \mathscr{D}'_{i-1})$;
6 $\quad$ **return** $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap C_{R'} \sqcap C \sqsubseteq D$;

If the special cases do not apply, we are left with the core case in Lines 4 – 6: the antecedent concept is exceptional and the problematic rank has more than one axiom. Sub-procedure `LAC` is used to "fine-comb" through the problematic rank $\mathcal{D}_{i-1}$. Starting with keeping $m - 1$ axioms from $\mathcal{D}_{i-1}$ (remember we cannot keep *all* m because $\{\mathcal{D}_n, \ldots, \mathcal{D}_i, \mathcal{D}_{i-1}\}$ is *not* C-compatible w.r.t. R), if this does not give C-compatibility then we keep $m - 2$ axioms and so on. We terminate when we reach C-compatibility or when $k = 0$ (we cannot keep any axioms). Consider the following example:

Example 26 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$ *where:*

$$\mathcal{T} = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \sqsubseteq & \mathsf{Student} \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{lcl} \mathsf{Student} & \mathrel{\sqsubset\!\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \mathrel{\sqsubset\!\!\!\sim} & \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{Student} & \mathrel{\sqsubset\!\!\!\sim} & \neg \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \mathrel{\sqsubset\!\!\!\sim} & \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \mathrel{\sqsubset\!\!\!\sim} & \exists \mathsf{receives.TaxInvoice}, \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \mathrel{\sqsubset\!\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{worksFor.University} & \mathrel{\sqsubset\!\!\!\sim} & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\}$$

□

Applying Procedure `ComputeRankingB` to $\langle \mathcal{T}, \mathcal{D} \rangle$ in Example 26 on Page 194 we obtain the ranking:

$$\mathcal{D}_0 = \left\{ \begin{array}{lcl} \mathsf{Student} & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{Student} & \precsim & \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{Student} & \precsim & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\},$$

$$\mathcal{D}_1 = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \precsim & \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \precsim & \exists \mathsf{receives.TaxInvoice} \end{array} \right\},$$

$$\mathcal{D}_2 = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{worksFor.University} & \precsim & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\}$$

Notice that $\mathsf{EmployedStudent} \precsim \exists \mathsf{access.UniversityLibrary}$ is not in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$. However, applying Procedure `LexicographicClosureA` to this query and KB gives the maximal EmployedStudent-compatible subset of the ranking as $\{\mathcal{D}_2, \mathcal{D}_1\}$, the problematic rank is $\mathcal{D}_0$. $\mathcal{D}'_k$ on Line 6 of Procedure `LAC` is calculated as $\{ \{ \mathsf{Student} \precsim \neg(\exists \mathsf{receives.TaxInvoice}) \}, \{ \mathsf{Student} \precsim \exists \mathsf{access.UniversityLibrary} \}, \{ \mathsf{Student} \precsim \neg \mathsf{SelfSponsoredStudent} \} \}$. Therefore, on Line 5 of Procedure `LexicographicClosureA`, $(\bigsqcup \mathcal{C}_{\mathcal{S} \in \mathcal{D}'_k}) = C_{R'} = (\neg \mathsf{Student} \sqcup \neg(\exists \mathsf{receives.TaxInvoice})) \sqcup (\neg \mathsf{Student} \sqcup \exists \mathsf{access.UniversityLibrary}) \sqcup (\neg \mathsf{Student} \sqcup \neg \mathsf{SelfSponsoredStudent})$. We observe that the middle disjunct is "compatible" with EmployedStudent and therefore $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqcap (\bigsqcup \mathcal{C}_{\mathcal{S} \in \mathcal{D}'_k}) \sqcap \mathsf{EmployedStudent} \sqsubseteq \exists \mathsf{access.UniversityLibrary}$.

Similar behaviour reveals that employed students who have children (as well as employed students who work for a university) can inherit the property of having access to a university library from general students, because of the fine-grained behaviour of Lexicographic Closure. We can also derive that employed students who have children are usually self sponsored, because this is a typical property of employed students. Another useful inference, that employed students who work for a university are obliged to pay taxes, can be made using Lexicographic Closure.

Hence, Lexicographic Closure does not inherit the inferential caution of Rational Closure described in Example 23 on Page 178.

We have to show that Procedure `LexicographicClosureA` terminates, and that it corresponds with the semantics presented in Section 4.4.1 (i.e., that the procedure is sound and complete). Termination is trivial to show because Procedure `Rank` and Sub-procedure `LAC` both terminate. The former has been shown earlier in this chapter, the latter can be extrapolated from Lines $3-5$ of Sub-procedure `LAC`.

Soundness and completeness for Procedure `LexicographicClosureA` can be shown by demonstrating that Definitions 39 and 43 on Pages 185 and 192 for Lexicographic Closure actually correspond. Before we do so, we prove some intermediate results. The first result shows that incoherence of a concept, w.r.t. a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, can only be caused by strict information. This means that, considering that $\langle \mathcal{T}, \mathcal{D} \rangle$ is LHS-coherent (all strict information is surely in $\mathcal{T}$), then incoherence of a concept can only be caused by $\mathcal{T}$ itself. Recall also that the case of a concept being incoherent w.r.t. a defeasible KB corresponds exactly to the concept having infinite rank. The following proves the result just discussed (and has been proven independently as well [41]).

Lemma 14 (Strict Facts are Responsible for Incoherence) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB and C a concept. Then, $\mathcal{T} \models C \sqsubseteq \bot$ if and only if $\langle \mathcal{T}, \mathcal{D} \rangle \models_r C \sqsubseteq \bot$.*

Proof: The contrapositive is $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r C \sqsubseteq \bot$ if and only if $\mathcal{T} \not\models C \sqsubseteq \bot$.

" $\Longrightarrow$ " Assume $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r C \sqsubseteq \bot$. This implies that there is a ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. there is an $x \in C^{\mathcal{R}}$. Let $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ s.t. $\Delta^{\mathcal{I}} = \Delta^{\mathcal{R}}$ and $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$. Assume that $\mathcal{I} \Vdash \mathcal{T}$. This means there is an $X \sqsubseteq Y \in \mathcal{T}$ s.t. $\mathcal{I} \nVdash X \sqsubseteq Y$ which, in turn, implies that there is a $y \in X^{\mathcal{I}}$ s.t. $y \notin Y^{\mathcal{I}}$. But this implies that $\mathcal{R} \nVdash X \sqsubseteq Y$ because $\Delta^{\mathcal{R}} = \Delta^{\mathcal{I}}$ and $\cdot^{\mathcal{R}} = \cdot^{\mathcal{I}}$. We reach a contradiction with $\mathcal{R} \Vdash \mathcal{T}$ and therefore, $\mathcal{I} \Vdash \mathcal{T}$ and it is clear that there is a $z \in C^{\mathcal{I}}$ corresponding to $x \in C^{\mathcal{R}}$. Hence, $\mathcal{T} \not\models C \sqsubseteq \bot$.
" $\Longleftarrow$ " Assume that $\mathcal{T} \not\models C \sqsubseteq \bot$. This means there is a model $\mathcal{I}$ for $\mathcal{T}$ s.t. there is an $x \in C^{\mathcal{I}}$. We also know that $\langle \mathcal{T}, \mathcal{D} \rangle$ is LHS-coherent which

implies that $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r X \sqsubseteq \bot$ for each $X \mathrel{\tilde{\sqsubseteq}} Y \in \mathcal{D}$. This obviously tells us that there is a ranked model $\mathcal{R}_X$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. $\mathcal{R}_X \nVdash X \sqsubseteq \bot$ for each $X \mathrel{\tilde{\sqsubseteq}} Y \in \mathcal{D}$. We pick such an $\mathcal{R}_X$ for each $X \mathrel{\tilde{\sqsubseteq}} Y \in \mathcal{D}$. Observe that, for each $\mathcal{R}_X$, by virtue of having an element $y \in X^{\mathcal{R}_X}$, there must be a *minimal* such element $y' \in X^{\mathcal{R}_X}$. This implies that y' will satisfy $X \mathrel{\tilde{\sqsubseteq}} Y \in \mathcal{D}$. We will refer to this observation later in the proof. For now, we take the horizontal disjoint union (Definition 27 on Page 132) of each $\mathcal{R}_X$ to obtain $\mathcal{R}'$. From Lemma 2 on Page 133 we know that $\mathcal{R}' \Vdash \langle \mathcal{T}, \mathcal{D} \rangle$. However, we do not know of any element in $\Delta^{\mathcal{R}'}$ that belongs to $C^{\mathcal{R}'}$ which is the goal of this proof. But we do know that there is an element $x \in C^{\mathcal{I}}$. Thus we plan to augment the ranked model $\mathcal{R}'$ with this information about $\mathcal{I}$. We define a ranked interpretation from the information in $\mathcal{I}$. That is, we define $\mathcal{R}_{\mathcal{I}} = \langle \Delta^{\mathcal{R}_{\mathcal{I}}}, \cdot^{\mathcal{R}_{\mathcal{I}}}, \prec_{\mathcal{R}_{\mathcal{I}}} \rangle$ s.t. $\Delta^{\mathcal{R}_{\mathcal{I}}} = \Delta^{\mathcal{I}}$, $\cdot^{\mathcal{R}_{\mathcal{I}}} = \cdot^{\mathcal{I}}$ and $\prec_{\mathcal{R}_{\mathcal{I}}}$ s.t. $rk_{\mathcal{R}_{\mathcal{I}}}(w) = 0$ for each $w \in \Delta^{\mathcal{R}_{\mathcal{I}}}$. Now, we take the vertical disjoint union (Definition 28 on Page 133) of $\mathcal{R}'$ and $\mathcal{R}_{\mathcal{I}}$ (i.e., the construction $\mathcal{R}' \boxminus \mathcal{R}_{\mathcal{I}}$) to obtain $\mathcal{R}''$. Since, by definition, we know there is an element of $C^{\mathcal{R}''}$, the crux is to show that $\mathcal{R}''$ is a ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$. It is easy to show that $\mathcal{R}'' \Vdash \mathcal{T}$ because $\mathcal{R}' \Vdash \mathcal{T}$, $\mathcal{R}_{\mathcal{I}} \Vdash \mathcal{T}$, $\Delta^{\mathcal{R}''} = \Delta^{\mathcal{R}'} \cup \Delta^{\mathcal{R}_{\mathcal{I}}}$ and $\cdot^{\mathcal{R}''} = \cdot^{\mathcal{R}'} \biguplus \cdot^{\mathcal{R}_{\mathcal{I}}}$ (Definition 28 on Page 133). To understand that $\mathcal{R}'' \Vdash \mathcal{D}$ it is sufficient to observe that any element of the topmost rank of $\mathcal{R}''$ (i.e., the elements inherited from $\mathcal{R}_{\mathcal{I}}$) satisfies $\mathcal{D}$ vacuously. This is because for any such element z, if $z \in X^{\mathcal{R}''}$ for some $X \mathrel{\tilde{\sqsubseteq}} Y \in \mathcal{D}$ then $z \notin \min_{\prec_{\mathcal{R}''}}(X^{\mathcal{R}''})$. This is because of the construction of $\mathcal{R}'$ which guarantees that there is an element of $X^{\mathcal{R}'}$ for each $X \mathrel{\tilde{\sqsubseteq}} Y \in \mathcal{D}$. And, because these elements are all of lower rank than z, z will always satisfy $\mathcal{D}$. Therefore, $\mathcal{R}'' \Vdash \langle \mathcal{T}, \mathcal{D} \rangle$ and there is a $z' \in C^{\mathcal{R}''}$. Hence, $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r C \sqsubseteq \bot$. □

The second result demonstrates that the minimal elements of a concept in the LCDM correspond exactly with those elements of the concept that satisfy the most number of subsumptions (materialised concepts) in the KB.

Lemma 15 (Minimality vs. Maximal Compatibility and LAC) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $\mathcal{R}$ its LCDM, $C \sqsubseteq\!\!\!\sim D$ a query, $R' = \{\mathcal{D}_n, \ldots, \mathcal{D}_i\}$ maximal C-compatible subset of its ranking $\{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$, and $C_{R'}$ the LAC for $\mathcal{D}_{i-1}$ w.r.t. R' and C. Then, $x \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$ if and only if $x \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^\mathcal{R}$ for any x.*

Proof: " $\Longrightarrow$ " Assume that $x \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$. Suppose the consequent of our proof statement holds (i.e., that $x \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^\mathcal{R}$). Then, by definition of $\mathcal{C}_{\bigcup R'} = \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i}$ and $C_{R'} = (\bigsqcup \mathcal{C}_{\mathcal{S} \in \mathcal{D}'_k})$, we know that the number of subsumptions $X \sqsubseteq\!\!\!\sim Y \in \mathcal{D}$ that x "satisfies" (see Definition 37 on Page 183) is defined as: the number of subsumptions in $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i$ *plus* the number of subsumptions in any $\mathcal{S} \in \mathcal{D}'_k$. We will refer to this number as ns_x. Hence, $ns_x = |\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i| + |\mathcal{S}|$ for any $\mathcal{S} \in \mathcal{D}'_k$. Now, assume that the consequent of our proof statement does *not* hold. I.e., assume that $x \notin (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^\mathcal{R}$. This means that either (1) there is an $X \sqsubseteq\!\!\!\sim Y \in \mathcal{D}_j$ for some $\mathcal{D}_j \in R'$ s.t. $x \in (X \sqcap \neg Y)^\mathcal{R}$, or (2) $x \notin (\bigsqcup \mathcal{C}_{\mathcal{S} \in \mathcal{D}'_k})^\mathcal{R}$ (or both). In either case, it will mean that $ns_x < |\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i| + |\mathcal{S}|$ for any $\mathcal{S} \in \mathcal{D}'_k$. Because we know that x is a minimal element of C in $\mathcal{R}$, we know that there is no $y \in C^\mathcal{R}$ s.t. $y \prec_\mathcal{R} x$. By definition of $\prec_\mathcal{R}$ (see Definition 38 on Page 185), we know that (i) there is no such y s.t. y is lower than x in the RCDM $\mathcal{R}_{rational}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ *and* (ii) there is no such y s.t. $ns_y > ns_x$. Observe that, by definition of maximal C-compatible set (Definition 36), our reduction to classical entailment for concept exceptionality (Theorem 1 on Page 143), and LAC (Definition 42) that $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqsubseteq \neg C$. Which is logically equivalent to saying that $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq \bot$. This means that there is a classical model $\mathcal{I}$ for $\mathcal{T}$ s.t. there is an element $x \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^\mathcal{I}$. From Lemma 14 on Page 196 we can derive that $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq \bot$. This implies that there is a ranked model $\mathcal{R}'$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. there is an element $z \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^{\mathcal{R}'}$. This last piece of information, together with the definition of minimal canonical ranked model, tells us that there is a $z' \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^\mathcal{R}$. That is, there is a representative element in $\mathcal{R}$ for each element in each standard ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$. But this would

mean that $ns_{z'} > ns_x$. We arrive at a contradiction with (ii). Therefore we have shown that $x \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^{\mathcal{R}}$.
" $\Longleftarrow$ " Assume that $x \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^{\mathcal{R}}$. Let $\mathcal{R}_{rational}$ be the RCDM for $\langle \mathcal{T}, \mathcal{D} \rangle$. By definition of maximal C-compatible subset $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i$ (Definition 36 on Page 172) and our reduction to classical entailment for concept exceptionality (Theorem 1 on Page 143), we know that $\langle \mathcal{T}, \mathcal{D}_n \cup \ldots \cup \mathcal{D}_i \rangle \not\models_r \top \sqsubseteq \neg C$, and that it is the smallest i for which this is possible. This tells us that the rank of C w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ (and also w.r.t. $\mathcal{R}_{rational}$) is i. That is, $rk_{\langle \mathcal{T}, \mathcal{D} \rangle} = rk_{\mathcal{R}_{rational}} = i$. Therefore, it is easy to see that x is a minimal element of C in $\mathcal{R}_{rational}$ because $\mathcal{R}_{rational}$ is a *minimal* ranked model. The goal is to show that after the lexicographic refinement of the ordering for $\mathcal{R}_{rational}$, x will remain a minimal element of C. By definition of $C_{R'} = (\bigsqcup \mathcal{C}_{\mathcal{S} \in \mathcal{D}'_k})$ we see that $|\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i| + |\mathcal{S}|$ for any $\mathcal{S} \in \mathcal{D}'_k$ is the *maximal* number of subsumptions in $\mathcal{D}$ that an element of C can satisfy. Therefore x satisfies the maximal number of subsumptions that any C can satisfy. Hence, according the Lexicographic refinement of the ordering (Definition 37 on Page 183), which favours ("pushes down") elements that satisfy more subsumptions from $\mathcal{D}$, x has to be a minimal element of C w.r.t. $\mathcal{R}$. That is, $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}})$.□

We can now move on to showing that our reduction to classical entailment for Lexicographic Closure, given by Procedure `LexicographicClosureA` on Page 194 and Definition 43 on Page 192, corresponds with our presented semantics for it (Definition 39 on Page 185). We formulate the following lemma to capture what we have to show.

Lemma 16 (Definitions 39 and 43 correspond) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $\mathcal{R}$ its LCDM, $C \sqsubseteq D$ a query, $R' = \{\mathcal{D}_n, \ldots, \mathcal{D}_i\}$ the maximal C-compatible subset of its ranking $\{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$, and $C_{R'}$ the LAC for $\mathcal{D}_{i-1}$ w.r.t. R' and C. Then, $\mathcal{T} \models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$ if and only if $\langle \mathcal{T}, \mathcal{D} \rangle \models_{lexico} C \sqsubseteq D$.*

Proof: " $\Longrightarrow$ " We have to show that if $\mathcal{T} \models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$ then $\langle \mathcal{T}, \mathcal{D} \rangle \models_{lexico} C \sqsubseteq D$. Since $\models_{lexico}$ is defined in terms of the LCDM $\mathcal{R}$, we

have to show that if $\mathcal{T} \models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$ then $\mathcal{R} \Vdash C \mathrel{\sqsubset\!\!\sim} D$. Assume that $\mathcal{T} \models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$ but that $\mathcal{R} \nVdash C \mathrel{\sqsubset\!\!\sim} D$. This means that there is an element $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}})$ s.t. $x \notin D^{\mathcal{R}}$. From Lemma 15 on Page 198 we know that $x \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^{\mathcal{R}}$. We obviously know that $\mathcal{R} \Vdash \mathcal{T}$. However, even stripping away the ordering component of $\mathcal{R}$ we are left with a classical interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ s.t. $\Delta^{\mathcal{I}} = \Delta^{\mathcal{R}}$, $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$ and, therefore, $\mathcal{I} \Vdash \mathcal{T}$. But, by our initial assumption that $\mathcal{T} \models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$ (each model for $\mathcal{T}$ satisfies $\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$), it follows that $\mathcal{I} \Vdash \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$. But we know that $x \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^{\mathcal{I}}$ (because $x \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C)^{\mathcal{R}}$, $\Delta^{\mathcal{I}} = \Delta^{\mathcal{R}}$ and $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$) and $x \notin D^{\mathcal{I}}$ (again because $x \notin D^{\mathcal{R}}$, $\Delta^{\mathcal{I}} = \Delta^{\mathcal{R}}$ and $\cdot^{\mathcal{I}} = \cdot^{\mathcal{R}}$). This is clearly a contradiction with $\mathcal{I} \Vdash \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$. Therefore, $\mathcal{R} \Vdash C \mathrel{\sqsubset\!\!\sim} D$.

" $\Longleftarrow$ " We have to show that if $\mathcal{R} \Vdash C \mathrel{\sqsubset\!\!\sim} D$ then $\mathcal{T} \models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$. Assume that $\mathcal{R} \Vdash C \mathrel{\sqsubset\!\!\sim} D$ but $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$. This is logically equivalent to saying that $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqcap \neg D \sqsubseteq \bot$. From Lemma 14 on Page 196 we know that $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqcap \neg D \sqsubseteq \bot$. Therefore, there is a ranked model $\mathcal{R}'$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. there is a $y \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqcap \neg D)^{\mathcal{R}'}$. This tells us that there is a $y' \in (\mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqcap \neg D)^{\mathcal{R}}$ by definition of minimal canonical ranked model (that is, there is a representative element in $\mathcal{R}$ for each element in each standard ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$). From Lemma 15 on Page 198 we know that $y' \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}})$ and we arrive at a contradiction with $\mathcal{R} \Vdash C \mathrel{\sqsubset\!\!\sim} D$ because $y' \notin D^{\mathcal{R}}$. Hence, $\mathcal{T} \models \mathcal{C}_{\bigcup R'} \sqcap C_{R'} \sqcap C \sqsubseteq D$. □

An analysis of Procedure `LexicographicClosureA` reveals that the worst-case computational complexity is increased from EXPTIME-COMPLETE (for classical $\mathcal{ALC}$) to 2-EXPTIME-COMPLETE (double exponential time). This can be demonstrated by observing that, in the worst case, our input defeasible KB will only contain defeasible axioms (i.e., $\mathcal{T} = \emptyset$). Therefore, to identify the LAC we need to compute an exponential number of subsets of size k of a set of n elements (where n is the size of the problematic rank). I.e., the number of disjuncts in the LAC would be exponential in the size of

the problematic rank. This exponentially-sized input for the $\mathcal{ALC}$ decision procedure, together with the exponential number of operations required (in the worst case) in the decision procedure itself, results in the 2-EXPTIME-COMPLETE complexity for Lexicographic Closure.

4.4.3 Discussion

We have given a semantics and procedure for computing the Lexicographic Closure of a defeasible KB. We have shown the correspondence between its ranked model semantics and its reduction to classical entailment. We have also demonstrated that Lexicographic Closure is an inferential extension of Rational Closure. Intuitively, Rational Closure defines a cautious inference mechanism ("I will only infer something if there is explicit evidence which proves it") and Lexicographic Closure defines the credulous counterpart to this: "I will infer something as long as there is no evidence to the contrary". We foresee that both inference mechanisms should have applicability in a variety of contexts. Perhaps Lexicographic Closure may be suitable for more real-world applications than Rational Closure because the latter may be viewed as defining an inference relation that is too cautious. We conclude this section by mentioning two aspects of Lexicographic Closure that point to possible variants which may also prove useful. The first aspect is that Lexicographic Closure is *syntax-dependent*. That is, performing Lexicographic Closure inference on two logically equivalent KBs (having the same ranked models but that are syntactically different) may yield different inferences. Consider the example:

Example 27 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$*:*

$$\mathcal{T} = \left\{ \mathsf{EmployedStudent} \sqsubseteq \mathsf{Student} \right\},$$

$$\mathcal{D} = \left\{ \begin{array}{lcl} \mathsf{Student} & \mathrel{\tilde{\sqsubseteq}} & \neg(\exists \mathsf{receives.TaxInvoice}) \sqcap \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{Student} & \mathrel{\tilde{\sqsubseteq}} & \neg \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \mathrel{\tilde{\sqsubseteq}} & \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{receives.TaxInvoice} \end{array} \right\}$$

□

Applying Procedure `ComputeRankingB` to $\langle \mathcal{T}, \mathcal{D} \rangle$ in Example 27 on Page 201 we obtain the ranking:

$$\mathcal{D}_0 = \left\{ \begin{array}{lcl} \mathsf{Student} & \precsim & \neg(\exists \mathsf{receives.TaxInvoice}) \sqcap \exists \mathsf{access.UniversityLibrary}, \\ \mathsf{Student} & \precsim & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\},$$

$$\mathcal{D}_1 = \left\{ \begin{array}{lcl} \mathsf{EmployedStudent} & \precsim & \mathsf{SelfSponsoredStudent}, \\ \mathsf{EmployedStudent} & \precsim & \exists \mathsf{receives.TaxInvoice} \end{array} \right\}$$

It is straightforward to verify that Lexicographic Closure does not give the intuitive inference that employed students have access to a university library. Recall from Example 26 on Page 194 that we obtained this inference with a logically equivalent KB. In that example, the axiom $\mathsf{Student} \precsim \neg(\exists \mathsf{receives.TaxInvoice}) \sqcap \exists \mathsf{access.UniversityLibrary}$ was represented in a more fine-grained manner. I.e., this axiom was "split" into the two axioms $\mathsf{Student} \precsim \neg(\exists \mathsf{receives.TaxInvoice})$ and $\mathsf{Student} \precsim \exists \mathsf{access.UniversityLibrary}$. Lexicographic Closure thus favours finer-grained representation of axioms in terms of giving back the most number of positive inferences.

The behaviour of Lexicographic Closure in Example 27 on Page 201 suggests possible variants of the procedure to catch these hidden inferences. That is, one could define possible normal forms for the KB which could "break axioms apart" into irreducible components, from which Lexicographic Closure would then be able to capture the hidden inferences. There have been numerous efforts concerning normal forms themselves in DLs, for applications other than the one we propose. One such related contribution is the work of Horridge, Parsia and Sattler [96] on laconic and precise *justifications* [94]. We anticipate that similar methods, to those used in this work, may be used to address the syntax sensitivity of Lexicographic Closure.

Finally, observing the behaviour of the Lexicographic Closure procedure, we notice that it defines an agnostic or naïve attitude towards identifying the maximal C-compatible knowledge. That is, it does not discriminate between the axioms to determine which ones contribute (and which ones don't) to the

exceptionality of the query antecedent. In other words, Lexicographic Closure defines a brute force approach to determining maximal C-compatibility, even though Procedure `LexicographicClosureA` is not a fully naïve procedure. Because of this behaviour, it is likely that Lexicographic Closure may not perform very efficiently in practice. Prima facie, an obvious solution is to identify heuristics to prune away many of the irrelevant axioms to the exceptionality in question. We now discuss a class of inference procedures related to Rational and Lexicographic Closure that adopt this approach.

4.5 Relevant Closure

As we demonstrated in Section 4.3, the Rational Closure is quite coarse in removing axioms from the ranking in order to reach C-compatibility. In fact, Rational Closure may eliminate axioms that are irrelevant to the exceptionality of the query antecedent. In the previous section we showed that Lexicographic Closure will not exhibit this behaviour. Rather, Lexicographic Closure is extremely naïve when its comes to eliminating axioms from the ranking that are irrelevant to the exceptionality of the query. It will eliminate a single axiom at a time until it reaches C-compatibility and so it is not in danger of eliminating irrelevant axioms. However, this brute force approach may prove to be inefficient from the perspective of reasoning performance.

In this section we present a class of reasoning procedures, called the *Relevant Closures* [48], that define a notion of *relevance* for axioms w.r.t. the exceptionality of the query antecedent. For all classes of Relevant Closure, relevance is defined in terms of *justifications* [94, 20]. A justification for an axiom that is entailed from a KB is a minimal subset (w.r.t. set inclusion) of the KB that entails the given axiom. In general there may be multiple justifications for an entailment. There has been extensive research into computing justifications for entailments in DL-based ontologies, and computing them efficiently [105, 94, 20]. Drawing from this work, we assume the existence of a black-box function called `AllJusts`$(\langle \mathcal{T}, \mathcal{D} \rangle, C)$ to compute the set of all

justifications for C being exceptional w.r.t. a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$. We give specialised definitions for justification and function $\texttt{AllJusts}(\langle \mathcal{T}, \mathcal{D} \rangle, C)$ for our purposes here.

Definition 44 (Justification for Exceptionality) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB and C a concept s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$. Then a* justification *for $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$ is a set $\mathcal{D}' \subseteq \mathcal{D}$ s.t. $\langle \mathcal{T}, \mathcal{D}' \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$ and there is no $\mathcal{D}'' \subset \mathcal{D}'$ s.t. $\langle \mathcal{T}, \mathcal{D}'' \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$. $\mathcal{D}'$ is also called a C-justification w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$.*

From this given definition of justification in our context of exceptionality, we can define $\texttt{AllJusts}(\langle \mathcal{T}, \mathcal{D} \rangle, C)$ as the function which returns the set of all C-justifications w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$. Since there are numerous optimised algorithms for $\texttt{AllJusts}(\langle \mathcal{T}, \mathcal{D} \rangle, C)$ in the literature we do not give a novel one here. In our subsequent algorithms for Relevant Closure, we assume the use of one of these existing procedures for computing all justifications.

In the remainder of this section we introduce three instances of the class of Relevant Closures. The first is known as *Basic Relevant Closure* (BRelC for short), the second as *Minimal Relevant Closure* (MRelC for short) and the third as *Lexicographically-Relevant Closure* (LRelC for short). Although all these constructions subscribe to a notion of *relevance* rooted in justifications, they differ slightly in how they apply justifications to identify maximal C-compatibility. We shall now focus on explicating the first instance of Relevant Closure called the Basic Relevant Closure.

Basic Relevant Closure defines the relevant axioms of $\langle \mathcal{T}, \mathcal{D} \rangle$ w.r.t. the exceptionality of C, as those that appear in some C-justification for $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$. I.e., the following definition of relevance is adopted.

Definition 45 (Relevance and C-basis) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, C a concept s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$, and $\mathcal{J} = \texttt{AllJusts}(\langle \mathcal{T}, \mathcal{D} \rangle, C) = \{\mathcal{J}_1, \ldots, \mathcal{J}_n\}$. Then, the C-*basis *for $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$ is the set $\mathcal{J}_1 \cup \ldots \cup \mathcal{J}_n$. An axiom $\alpha \in \mathcal{D}$ is* relevant *for $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$ if $\alpha \in \mathcal{J}_1 \cup \ldots \cup \mathcal{J}_n$.*

That is, Basic Relevant Closure views any axiom appearing in a C-justification as relevant to the exceptionality of C. The basic idea, then, is to only eliminate axioms that appear in this set from the ranking when identifying the maximal C-compatible set. In order to capture this new behaviour, we will introduce an adaptation (to the case of Basic Relevant Closure) of the notion of maximal C-compatibility (Definition 36 on Page 172). We call this notion *maximally-relevant* C-compatibility.

Definition 46 (Maximally-relevant C-compatibility) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its ranking, $C \sqsubseteq\!\!\!\sim D$ a query s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$, and $\mathscr{C}$ the C-basis w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq\!\!\!\sim \neg C$. Now, let R' be a sequence of subsets $\mathcal{D}'_0, \ldots, \mathcal{D}'_i$ where $\mathcal{D}'_j = \mathcal{D}_j \backslash \mathscr{C}$ for $0 \leq j \leq i$ and let $R'' \subset R$ be the sequence of subsets $\mathcal{D}_{i+1}, \ldots, \mathcal{D}_n$. Then, $R' \cup R''$ (resp. $\mathcal{C}_{\bigcup(R' \cup R'')}$) is* maximally-relevant C*-compatible w.r.t. R if $\mathcal{T} \cup \mathcal{D}_n \cup \ldots \cup \mathcal{D}'_i \cup \ldots \cup \mathcal{D}'_0 \not\models_r \top \sqsubseteq\!\!\!\sim \neg C$ and there is no $k < i$ s.t. $\mathcal{T} \cup \mathcal{D}_n \cup \ldots \cup \mathcal{D}'_k \cup \ldots \cup \mathcal{D}'_0 \not\models_r \top \sqsubseteq\!\!\!\sim \neg C$.*

Intuitively, the maximally-relevant C-compatible set of a ranking characterises a procedure for computing maximal C-compatibility, where at each step we only consider eliminating *relevant* axioms from the ranking (i.e., those in $\mathcal{D}_j \cap \mathscr{C}$). We can now easily formalise Basic Relevant Closure in terms of classical entailment.

Definition 47 (Basic Relevant Closure of a Defeasible KB) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its ranking, $C \sqsubseteq\!\!\!\sim D$ a query, and R' the maximally-relevant C-compatible subset of R. Then, $C \sqsubseteq\!\!\!\sim D$ is in the* Basic Relevant Closure *of $\langle \mathcal{T}, \mathcal{D} \rangle$ if $\mathcal{T} \models \mathcal{C}_{\bigcup R'} \sqcap C \sqsubseteq D$.*

We do not give a ranked model semantics here for Basic Relevant Closure. One of the reasons is that Basic Relevant Closure does not characterise a rational consequence relation. That is, it does not satisfy some of the KLM postulates and therefore a ranked model semantics is, perhaps, less interesting for such a consequence relation. Even though this is an unfortunate

theoretical result, it does not detract from Basic Relevant Closure possibly having useful applications. We will give more detail about this in Chapter 5. In fact, in that chapter we show that most defeasible reasoning formalisms mentioned in this thesis do not natively characterise rational consequence relations.

For now we still have to provide a procedure for Basic Relevant Closure. Just like Lexicographic Closure, we use the same basic template of the Rational Closure algorithm but substitute our augmented notion of maximal C-compatibility. Pseudocode is presented in Procedure `BasicRelevantClosure`.

Procedure `BasicRelevantClosure`$(\langle \mathcal{T}, \mathcal{D} \rangle, R, \mathscr{C}, \delta)$

Input: A LHS-coherent defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, its ranking $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$, the C-basis $\mathscr{C}$, and a query $\delta = C \sqsubset\!\!\sim D$.

Output: **true** if δ is in the Basic Relevant Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$, **false** otherwise.

1 $i := 0$;
2 $R'' := R$;
3 **while** $\mathcal{T} \models \mathcal{C}_{\bigcup R''} \sqsubseteq \neg C$ **do**
4 **if** $i = n$ **then**
5 **return** $\mathcal{T} \models C \sqsubseteq D$;
6 $R'' := R''\backslash(\mathcal{D}_i \cap \mathscr{C})$;
7 $i := i + 1$;
8 **return** $\mathcal{T} \models \mathcal{C}_{\bigcup R''} \sqcap C \sqsubseteq D$;

Procedure `BasicRelevantClosure` works by following the same basic sequence of Rational Closure, except, at each stage where axioms are eliminated from the ranking, only *relevant* axioms are eliminated (Line 6). Observe that we never have to eliminate any other axioms because they are irrelevant to the exceptionality of the query. I.e., the axioms in $\mathcal{D}\backslash\mathscr{C}$ are always retained (see Line 3). We illustrate the behaviour of Procedure `BasicRelevantClosure` with an example.

Example 28 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$*:*

$$\mathcal{T} = \left\{ \begin{array}{lll} 1.\ \mathsf{EmployedStudent} & \sqsubseteq & \mathsf{Student} \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{lll} 2.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ 3.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{access.UniversityLibrary}, \\ 4.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg \mathsf{SelfSponsoredStudent}, \\ 5.\ \mathsf{EmployedStudent} & \mathrel{\sqsubset\!\!\sim} & \mathsf{SelfSponsoredStudent}, \\ 6.\ \mathsf{EmployedStudent} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{receives.TaxInvoice}, \\ 7.\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ 8.\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{worksFor.University} & \mathrel{\sqsubset\!\!\sim} & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\}$$

□

Applying Procedure `ComputeRankingB` to $\langle \mathcal{T}, \mathcal{D} \rangle$ in Example 28 on Page 206 we obtain the ranking:

$$\mathcal{D}_0 = \left\{ \begin{array}{lll} 2.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ 3.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{access.UniversityLibrary}, \\ 4.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\},$$

$$\mathcal{D}_1 = \left\{ \begin{array}{lll} 5.\ \mathsf{EmployedStudent} & \mathrel{\sqsubset\!\!\sim} & \mathsf{SelfSponsoredStudent}, \\ 6.\ \mathsf{EmployedStudent} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{receives.TaxInvoice} \end{array} \right\},$$

$$\mathcal{D}_2 = \left\{ \begin{array}{lll} 7.\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ 8.\ \mathsf{EmployedStudent} \sqcap \exists \mathsf{worksFor.University} & \mathrel{\sqsubset\!\!\sim} & \neg \mathsf{SelfSponsoredStudent} \end{array} \right\}$$

Suppose our query is the axiom $\mathsf{EmployedStudent} \mathrel{\sqsubset\!\!\sim} \exists \mathsf{access.UniversityLibrary}$. The EmployedStudent-justifications are $\{1, 2, 6\}$ and $\{1, 4, 5\}$. Therefore, the EmployedStudent-basis is $\{1, 2, 4, 5, 6\}$. On the first iteration of the while loop in Procedure `BasicRelevantClosure` we eliminate Axiom 2 and Axiom 4. We do not need to eliminate any more axioms because we have EmployedStudent-compatibility. It is easy to see that $\mathsf{EmployedStudent} \mathrel{\sqsubset\!\!\sim} \exists \mathsf{access.UniversityLibrary}$ is indeed in the Basic Relevant Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$. Recall that Rational Closure would not give back this inference. So BRelC does not have the same inferential caution that Rational Closure has.

However, Basic Relevant Closure can also be seen to be too "coarse" in its elimination of axioms. Observe that $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \mathrel{\tilde{\sqsubseteq}} \mathsf{SelfSponsoredStudent}$ cannot be inferred. But it seems intuitive to be able to conclude this seeing as $\mathsf{EmployedStudent} \mathrel{\tilde{\sqsubseteq}} \mathsf{SelfSponsoredStudent}$, and there is nothing which precludes one from inferring that employed students who also have children are usually self sponsored. The reason for this caution is that Basic Relevant Closure does not distinguish among the axioms appearing in the $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$-basis, when eliminating them from a particular rank. I.e., it removes *all* such axioms from that rank.

In our example the $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$-justifications are $\mathcal{J}_1 = \{1, 2, 6\}$, $\mathcal{J}_2 = \{1, 4, 5\}$ and $\mathcal{J}_3 = \{6, 7\}$. The $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$-basis is therefore $\{1, 2, 4, 5, 6, 7\}$. On the first iteration of the while loop in the procedure we remove Axiom 2 and Axiom 4 as usual. As a matter of interest, notice that Axioms 2 and Axiom 4 are the lowest ranked axioms in $\mathcal{J}_1$ and $\mathcal{J}_2$ respectively. On the second iteration we will remove *both* Axiom 5 and Axiom 6 from the ranking (because they both appear in the $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$-basis). However, notice that Axiom 5 is not the lowest ranked axiom in any $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$-justification.

Recalling that Rational Closure removes axioms in order from lowest ranked to higher ranked, intuition tells us that applying a similar behaviour to Relevant Closures would give more sound inferences. That is, perhaps we should choose our $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$-basis as the lowest ranked axioms from each of the $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top$-justifications. In fact, this behaviour allows us to infer $\mathsf{EmployedStudent} \sqcap \exists \mathsf{hasChild}.\top \mathrel{\tilde{\sqsubseteq}} \mathsf{SelfSponsoredStudent}$ from our example because we will not remove Axiom 5 from $\mathcal{D}_1$ (only Axiom 6 will be removed).

Because of the finite number of ranks, Procedure `BasicRelevantClosure` will terminate (n is finite). The soundness and completeness (correspondence with Definition 47 on Page 205) follows straightforwardly by noticing that $\mathcal{C}_{\bigcup R''}$ (Line 8 of Procedure `BasicRelevantClosure` on Page 206 corresponds exactly with $\mathcal{C}_{\bigcup R'}$ in Definition 47 on Page 205 and $\mathcal{C}_{\bigcup (R' \cup R'')}$ in Definition 46

on Page 205. The computational complexity of Basic Relevant Closure increases to 2-EXPTIME-COMPLETE just like Lexicographic Closure. This is because we require computation of all C-justifications which has been shown to require an exponential number of entailment checks in the number of defeasible subsumptions in the KB [94]. Specifically, computation of all justifications is accomplished by constructing a *hitting set tree* [162] where each node in the tree is labelled with a justification for the entailment in question.

The basic procedure for constructing this graph assumes that we have a procedure or black-box for computing *one* justification for the entailment in question. Using such a procedure we generate the root node of the tree. Then, for each axiom α in the label of this node, we construct an edge (labelled with α) from the root to a newly constructed node, say n. The label for a node n will then be a new justification for the entailment generated using our black-box procedure, with the input ontology minus all the axioms on the *path* (labels on the sequence of edges from the root node) to n. If the entailment no longer holds at some node n then that branch of the tree is "closed" with the terminating label 'x'. The construction is complete when all leaves of the tree are labelled with 'x'.

From the above construction we notice that when none of the justifications overlap (or share any elements) this number of nodes in this tree grows exponentially in the worst case (in the number of justifications for the entailment).

Returning to Example 28 on Page 206, recall that we discussed a slightly different notion of relevance that one could define in terms of C-justifications. That is, we could restrict ourselves to the lowest ranked axioms in each C-justification (as opposed to keeping them all). This would give us a principled and finer-grained behaviour when eliminating axioms from the ranking. This is the philosophy adopted by the second instance of Relevant Closure, called *Minimal Relevant Closure*. The Minimal Relevant Closure procedure follows the same structure as that of Basic Relevant Closure. The only difference is in its notion of relevance which is based on the minimally ranked axioms in

the C-justifications. This notion of relevance is defined by *minimal-relevance* and *minimal* C-basis .

Definition 48 (Minimal Relevance and Minimal C-basis) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, C a concept s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq \neg C$, and $\mathcal{J} = \texttt{AllJusts}(\langle \mathcal{T}, \mathcal{D} \rangle, C) = \{\mathcal{J}_1, \ldots, \mathcal{J}_n\}$. Then, the* minimal C-basis *for $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq \neg C$ is the set $\mathcal{J}'_1 \cup \ldots \cup \mathcal{J}'_n$ where $\mathcal{J}'_i = \{\alpha \in \mathcal{J}_i \mid$ there is no $\alpha' \in \mathcal{J}_i$ s.t. $rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(\alpha') < rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(\alpha)\}$. An axiom $\beta \in \mathcal{D}$ is* minimally relevant *for $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \sqsubseteq \neg C$ if $\beta \in \mathcal{J}'_1 \cup \ldots \cup \mathcal{J}'_n$.*

Given this new notion of relevance, the maximally-relevant C-compatible subset of the ranking (Definition 46 on Page 205) is analogously defined for Minimal Relevant Closure. The only difference is that instead of C-basis in Definition 46, we use the notion of minimal C-basis (Definition 48 on Page 210). The same can be said of the definition for Minimal Relevant Closure in terms of classical entailment. It is analogous to Basic Relevant Closure. The only difference being that the notion of relevance subscribed to is defined by minimal C-basis which, in turn, defines a slightly different notion of maximally-relevant C-compatibility. Therefore, the pseudocode below is given for completeness even though it is virtually the same as Procedure `BasicRelevantClosure` on Page 206. The only difference is that the minimal C-basis is supplied as input (rather than the standard C-basis).

Termination, soundness and completeness for `MinimalRelevantClosure` on Page 211 follow the same arguments as Procedure `BasicRelevantClosure` on Page 206. Furthermore, the computational complexity, yet again, remains 2-EXPTIME-COMPLETE as is the case for Basic Relevant Closure because we do not do any additional operations. I.e., we only restrict ourselves to a subset of the C-basis based on the ranks of the axioms (which can be "read-off" directly from R). As we have shown in Example 28 on Page 206, Minimal Relevant Closure is less cautious than Basic Relevant Closure and captures inferences that Basic Relevant Closure cannot. In other words, Minimal Relevant Closure is an inferential extension of Basic Relevant Closure. We demonstrate this below.

Procedure `MinimalRelevantClosure`$(\langle \mathcal{T}, \mathcal{D} \rangle, R, \mathscr{C}, \delta)$

Input: A LHS-coherent defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, its ranking $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$, the minimal C-basis $\mathscr{C}$, and a query $\delta = C \sqsubseteq\!\!\!\sim D$.

Output: **true** if δ is in the Minimal Relevant Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$, **false** otherwise.

```
1 i := 0;
2 R'' := R;
3 while T ⊨ C_{⋃R''} ⊑ ¬C do
4     if i = n then
5         return T ⊨ C ⊑ D;
6     R'' := R''\(D_i ∩ 𝒞);
7     i := i + 1;
8 return T ⊨ C_{⋃R''} ⊓ C ⊑ D;
```

Lemma 17 (MRelC Extends BRelC) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB and $C \sqsubseteq\!\!\!\sim D$ a query. Then, if $C \sqsubseteq\!\!\!\sim D$ is in the Basic Relevant Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$ then $C \sqsubseteq\!\!\!\sim D$ is in the Minimal Relevant Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$.*

Proof: We pick a $C \sqsubseteq\!\!\!\sim D$ in the Basic Relevant Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$. Let $\mathscr{C}$ be the C-basis w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ and $\mathscr{C}_{min}$ the minimal C-basis w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$. It is clear by their definitions (Definition 45 on Page 204 and 48 on Page 210) that $\mathscr{C}_{min} \subseteq \mathscr{C}$. Notice that, by Definition 46 on Page 205, the maximally relevant C-compatible subset defined in terms of the *minimal* C-basis - $\mathscr{C}_{min}$, will be a superset of the maximally relevant C-compatible subset defined in terms of the standard C-basis - $\mathscr{C}$. By monotonicity of classical entailment and Definition 47 on Page 205 it follows that $C \sqsubseteq\!\!\!\sim D$ has to be in the Minimal Relevant Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$. □

We will not give a ranked model semantics for Minimal Relevant Closure either because it is not a rational consequence relation. However, at this

point, the reader may enquire about the significance or logical merit of the Relevant Closures since they are not rational consequence relations. What we can say is that Basic and Minimal Relevant Closures do not satisfy *all* the KLM postulates but they do satisfy most of the foundational ones which may give credence to their logical merit. We will give more details about the satisfaction of postulates in Chapter 5. For now we can link Basic Relevant Closure to Rational Closure in terms of inferential power [48, Proposition 2].

Lemma 18 (BRelC Extends Rational Closure) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its ranking, $C \sqsubset\!\!\!\sim D$ a query and $\mathscr{C}$ is the C-basis w.r.t. R. If $C \sqsubset\!\!\!\sim D$ is in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$ then $C \sqsubset\!\!\!\sim D$ is in the Basic Relevant Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$.*

Proof: We know from Lemmas 11 and 12 that: $C \sqsubset\!\!\!\sim D$ is in the Rational Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$ if and only if $\mathcal{T} \models \mathcal{C}_{\bigcup MC} \sqcap C \sqsubseteq D$, where MC is the maximally C-compatible subset of R (see Definition 36 on Page 172). We also know from Definition 47 on Page 205 that $C \sqsubset\!\!\!\sim D$ is in the Basic Relevant Closure of $\langle \mathcal{T}, \mathcal{D} \rangle$ if and only if $\mathcal{T} \models \mathcal{C}_{\bigcup MRC} \sqcap C \sqsubseteq D$ where MRC is the maximally-relevant C-compatible subset of R (see Definition 46 on Page 205). The crux of this proof is to show that $MC \subseteq MRC$ and by monotonicity of classical entailment the result stated in this lemma would follow immediately. Observe that the special case where C is not exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$ implies that $\mathscr{C} = \emptyset$ and this culminates in a case where $MC = MRC$. It is clear that in such a case, Rational Closure and Basic Relevant Closure will correspond. We instead focus on the core case where C is exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$.

We begin by unpacking the definitions of MC and MRC. MC is the set $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i$ where $0 \leq i \leq n$ is the smallest number s.t. $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqsubseteq \neg C$. MRC is the set $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_{j+1} \cup \mathcal{D}'_j \cup \ldots \cup \mathcal{D}'_0$ where $0 \leq j \leq n$ is the smallest number s.t. $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_{j+1} \cup \mathcal{D}'_j \cup \ldots \cup \mathcal{D}'_0} \sqsubseteq \neg C$ (where $\mathcal{D}'_k = \mathcal{D}_k \backslash \mathscr{C}$ for $0 \leq k \leq n$). There are three cases:
Case 1: $i = j$. By definition of MRC we know that $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_{j+1} \cup \mathcal{D}'_j \cup \ldots \cup \mathcal{D}'_0} \sqsubseteq \neg C$. This means that $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j \cup \mathcal{D}'_{j-1} \cup \ldots \cup \mathcal{D}'_0} \sqsubseteq \neg C$. Because of our

assumption in this case that $i = j$ and that $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqsubseteq \neg C$ (by definition of MC), we know that $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j} \sqsubseteq \neg C$. Therefore, it must be the case that $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}'_j} \sqsubseteq \neg C$ because of classical monotonicity and the fact that $\mathcal{D}'_j \subseteq \mathcal{D}_j$ ($\mathcal{D}'_j = \mathcal{D}_i \backslash \mathscr{C}$). Hence, there must exist an $\alpha \in \mathcal{D}'_{j-1} \cup \ldots \cup \mathcal{D}'_0$ s.t. $\alpha \in \mathscr{C}$ (so that we can obtain a C-justification in $\mathcal{T} \cup \mathcal{D}_n \cup \ldots \mathcal{D}_{j+1} \cup \mathcal{D}'_j \cup \ldots \cup \mathcal{D}'_0$). But this is impossible because, by definition of $\mathcal{D}'_k$ we know that there is no $\alpha \in \mathcal{D}'_{j-1} \cup \ldots \cup \mathcal{D}'_0$ s.t. $\alpha \in \mathscr{C}$. Hence, we arrive at a contradiction.
Case 2: $i < j$. By definition of MC we know that $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i} \sqsubseteq \neg C$. But from $i < j$ we also know that $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j \subseteq \mathcal{D}_n \cup \ldots \cup \mathcal{D}_i$. By classical monotonicity this means that $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j} \sqsubseteq \neg C$. This also means that $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}'_j} \sqsubseteq \neg C$ because $\mathcal{D}'_j \subseteq \mathcal{D}_j$ (by definition of $\mathcal{D}'_k$ for $0 \leq k \leq n$). By definition of MRC it is clear that $\mathcal{T} \models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j \cup \mathcal{D}'_{j-1} \cup \ldots \mathcal{D}'_0} \sqsubseteq \neg C$ (this is because j is the smallest number s.t. $\mathcal{T} \not\models \mathcal{C}_{\mathcal{D}_n \cup \ldots \cup \mathcal{D}_{j+1} \cup \mathcal{D}'_j \cup \ldots \mathcal{D}'_0} \sqsubseteq \neg C$). Therefore, there must exist an $\alpha \in \mathcal{D}'_{j-1} \cup \ldots \cup \mathcal{D}'_0$ s.t. $\alpha \in \mathscr{C}$ (in order to obtain a C-justification in $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_j \cup \mathcal{D}'_{j-1} \cup \ldots \mathcal{D}'_0$). But this is impossible by definition of $\mathcal{D}'_k$ for $0 \leq k \leq n$. Hence, we arrive at a contradiction.
Case 3: $i > j$. This is the only possible case. By definition we have $MC = \mathcal{D}_n \cup \ldots \cup \mathcal{D}_i$ and $MRC = \mathcal{D}_n \cup \ldots \cup \mathcal{D}_{j+1} \cup \mathcal{D}'_j \cup \ldots \mathcal{D}'_0$. And because of our assumption that $i > j$ it is clear that $i \geq j + 1$ and therefore that $\mathcal{D}_n \cup \ldots \cup \mathcal{D}_i \subseteq \mathcal{D}_n \cup \ldots \cup \mathcal{D}_{j+1} \cup \mathcal{D}'_j \cup \ldots \mathcal{D}'_0$ (and hence $MC \subseteq MRC$). □

In other words, Lemma 18 on Page 212 says that Basic Relevant Closure is inferentially stronger than Rational Closure and from Lemma 17 on Page 211 we know that Minimal Relevant Closure is inferentially stronger than Basic Relevant Closure. Of course, the right to left directions of these statements will not hold in general. Therefore, since Rational Closure has a solid and well-motivated characterisation both semantically and proof-theoretically, its inferential relationship with Basic and Minimal Relevant Closure provides good argument in favour of the sensibility of the latter constructions (as useful defeasible reasoning methodologies).

We will now discuss the last known instance of Relevant Closure called the *Lexicographically-Relevant Closure*. This instance of Relevant Closure gets its name from the fact that it is actually a possible optimisation for computing Lexicographic Closure, using the notion of relevance adopted by Relevant Closures. Consider the following example.

Example 29 *Consider the following defeasible KB* $\langle \mathcal{T}, \mathcal{D} \rangle$*:*

$$\mathcal{T} = \left\{ \begin{array}{lcl} \textit{1.}\ \mathsf{GradStudent} & \sqsubseteq & \mathsf{Student}, \\ \textit{2.}\ \mathsf{ResearchStudent} & \equiv & \mathsf{GradStudent} \sqcap \forall \mathsf{enrolledIn.PureResearchDegree} \end{array} \right\},$$

$$\mathcal{D} = \left\{ \begin{array}{lcl} \textit{3.}\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ \textit{4.}\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{access.UniversityLibrary}, \\ \textit{5.}\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{assigned.OfficeSpace}), \\ \textit{6.}\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{takes.Course}, \\ \textit{7.}\ \mathsf{ResearchStudent} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{takes.Course}), \\ \textit{8.}\ \mathsf{ResearchStudent} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{assigned.OfficeSpace} \end{array} \right\}$$

□

Applying Procedure `ComputeRankingB` to $\langle \mathcal{T}, \mathcal{D} \rangle$ in Example 29 on Page 214 we obtain the ranking:

$$\mathcal{D}_0 = \left\{ \begin{array}{lcl} 3.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{receives.TaxInvoice}), \\ 4.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{access.UniversityLibrary}, \\ 5.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{assigned.OfficeSpace}), \\ 6.\ \mathsf{Student} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{takes.Course}, \end{array} \right\},$$

$$\mathcal{D}_1 = \left\{ \begin{array}{lcl} 7.\ \mathsf{ResearchStudent} & \mathrel{\sqsubset\!\!\sim} & \neg(\exists \mathsf{takes.Course}), \\ 8.\ \mathsf{ResearchStudent} & \mathrel{\sqsubset\!\!\sim} & \exists \mathsf{assigned.OfficeSpace} \end{array} \right\}$$

Suppose our query is the axiom $\mathsf{ResearchStudent} \mathrel{\sqsubset\!\!\sim} \exists \mathsf{access.UniversityLibrary}$. Lexicographic Closure will answer affirmatively to this query. The maximal ResearchStudent-compatible subset of the ranking is $\mathcal{D}_1$. The LAC for ResearchStudent w.r.t. the ranking is $(\bigsqcap\{3,4\}) \sqcup (\bigsqcap\{3,5\}) \sqcup (\bigsqcap\{3,6\}) \sqcup (\bigsqcap\{4,5\}) \sqcup (\bigsqcap\{4,6\}) \sqcup (\bigsqcap\{5,6\})$. In other words, all ways of keeping two

axioms from $\mathcal{D}_0$ results in maximal ResearchStudent-compatibility. It is easy to see that the first disjunct of the LAC is compatible with ResearchStudent (together with $\mathcal{D}_1$) and therefore the LAC as a whole is compatible with ResearchStudent. However, note that according to Procedure `LAC`, we have to perform two operations to arrive at the LAC. That is, we first have to compute all ways of keeping *three* axioms and check if this is compatible with ResearchStudent. It turns out that this is not the case, therefore we have to compute all ways of keeping two axioms (which is compatible). Hence, there will be a large number of such iterations when the size of the problematic rank is large.

Recall that, for this procedure, if our problematic rank has n axioms then we require at most $n - 1$ combinatorial computations to arrive at the LAC. To avoid a potential bottleneck in performance, and assuming that one has computed the C-basis for our query, we can make use of it to avoid many of the iterations of Procedure `LAC`. In this case the ResearchStudent-basis $\mathscr{C} = \{5, 6, 7, 8\}$ (remember that we assume Axioms 1 and 2 are background knowledge since they are strict, and therefore cannot be altered or eliminated). How many iterations can we avoid at the very least? Well, we can observe that $\mathcal{D}_0 \backslash \mathscr{C}$ will return the set of axioms from $\mathcal{D}_0$ that do not appear in a ResearchStudent-justification, and this means that these axioms are irrelevant to the exceptionality of ResearchStudent. In other words, we know we always have to keep these axioms and hence we save $|\mathcal{D}_0 \backslash \mathscr{C}|$ iterations. Therefore, we only have to perform Procedure `LAC` on the set $\mathcal{D}_0 \cap \mathscr{C}$ with $\mathcal{D}_0 \backslash \mathscr{C}$ (the irrelevant axioms) as background knowledge at each step.

Applied to our example we notice that $\mathcal{D}_0 \backslash \mathscr{C} = \{3, 4\}$ and so we will always keep these axioms. We then apply Procedure `LAC` with $\mathcal{D}_0 \cap \mathscr{C} = \{5, 6\}$ as the problematic rank (with $\{3, 4\}$ included in our background knowledge at each step). We try all ways of keeping one axiom from this set which cannot be achieved since both Axiom 5 and 6 are sufficient to cause exceptionality of ResearchStudent (together with $\mathcal{D}_1$). Therefore, we have to eliminate both Axiom 5 and 6 to obtain ResearchStudent-compatibility. We thus use only

one combinatorial computation for calculating the LAC, as opposed to two without the helpful C-basis optimisation. Note that in the worst case for our example, we would have required three combinatorial computations to derive the LAC ($|\mathcal{D}_0| = 4$).

Of course, we need to formalise our description of Lexicographically-Relevant Closure given in the above discussion. The definition and procedure for this closure will be analogous to those for Lexicographic Closure (see Definition 43 on Page 192 and Procedure `LexicographicClosureA` on Page 194). The only difference is that Lexicographically-Relevant Closure will conform to a slightly different notion of LAC (see Definition 42 on Page 191). We call this new notion the *lexicographically-relevant additive concept* (LRAC for short).

Definition 49 (Lexicographically Relevant Additive Concept) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its ranking, $C \sqsubset\!\!\!\sim D$ a query, $R' = \{\mathcal{D}_n, \ldots, \mathcal{D}_i\}$ the maximal C-compatible subset of R, $\mathscr{C}$ the C-basis for $\langle \mathcal{T}, \mathcal{D} \rangle$, $\mathcal{D}_{relevant} = \mathcal{D}_{i-1} \cap \mathscr{C}$, $\mathcal{D}_{irrelevant} = \mathcal{D}_{i-1} \backslash \mathscr{C}$ and $\mathscr{D}_{relevant} = \{\mathcal{D}'_1, \ldots, \mathcal{D}'_m\}$ the Lexicographicalisation of $\mathcal{D}_{relevant}$. Let k be the largest number s.t. $1 \leq k \leq m-1$ and $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap (\mathcal{C}_{\mathcal{D}_{irrelevant}}) \sqcap (\bigsqcup \mathcal{C}_{\mathcal{S} \in \mathcal{D}'_k}) \sqsubseteq \neg C$. Then, the concept $(\mathcal{C}_{\mathcal{D}_{irrelevant}}) \sqcap (\bigsqcup \mathcal{C}_{\mathcal{S} \in \mathcal{D}'_k})$ is known as the* lexicographically-relevant additive concept *(LRAC) for C w.r.t. R. $\mathcal{D}'_k$ is called the* lexicographically-relevant additive subset *(LRAS) of R w.r.t. C.*

The attentive reader will notice that showing the exact correspondence between LAC and LRAC proves that Lexicographically-Relevant Closure and Lexicographic Closure characterise the same inferences. Lexicographically-Relevant Closure is thus a potential optimisation for computing Lexicographic Closure. However, in order to show the correspondence between the LAC and LRAC, we have to unpack their definitions which are admittedly quite dense. We shall restrict our attention to the notions of LAS and LRAS described in the Definitions for LAC and LRAC. The reason is that we wish to strip away the internalisation mechanism of LAC and LRAC to

make our proofs easier to follow. The crux of the matter is then to show that the LAS and LRAS correspond exactly. Our strategy for showing this is to represent the computation of the LAS and LRAS as *tree problems* and show that the solutions of these problems correspond exactly.

Definition 50 (Lexicographic-tree) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB and $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its ranking. A* Lexicographic-tree *for the problematic rank $\mathcal{D}_{i-1}$ is an edge-labelled and node-labelled tree $T = \langle N, E, L_n, L_e \rangle$ where N is a set of nodes, E is a set of edges, L_n is a node labelling function and L_e is an edge labelling function s.t.:*

1. $L_n : N \to \mathcal{P}(\mathcal{D}_{i-1})$ where $\mathcal{P}(\mathcal{D}_{i-1})$ is the powerset of $\mathcal{D}_{i-1}$ and $L_e : E \to \mathcal{D}_{i-1}$.
2. The root node $r \in N$ of T is s.t. $L_n(r) = \mathcal{D}_{i-1}$.
3. For each $n_1, n_2 \in N$, there is an edge $e_{n_1 \to n_2} \in E$ from n_1 to n_2 if $L_n(n_2) \subset L_n(n_1)$ and there is no $n_3 \in N$ s.t. $L_n(n_2) \subset L_n(n_3) \subset L_n(n_1)$.
4. For each $e_{n_1 \to n_2} \in E$, $L_e(e_{n_1 \to n_2}) = L_n(n_1) \backslash L_n(n_2)$.
5. For some $n \in N$, $Path(n)$ denotes the set of edge labels from r to n.

Notice that the label of each edge in T represents a single axiom from $\mathcal{D}_{i-1}$. We also point out that $L_n(n_1) = L_n(r) \backslash Path(n_1)$ for any $n_1 \in N$. Finally, observe that T has a single leaf node l s.t. $L_n(l) = \emptyset$.

Therefore, Definition 50 on Page 217 describes a tree in which the root node corresponds to the full set of axioms in the problematic rank $\mathcal{D}_{i-1}$. For each axiom $\alpha \in \mathcal{D}_{i-1}$ we can remove α from this set to obtain a subset of $\mathcal{D}_{i-1}$ which will represent a new node in the tree. For each of these subsets, in turn, we can remove a single axiom from them to obtain subsets of these and so on. We can follow this construction until we have a node for each axiom in $\mathcal{D}_{i-1}$ (i.e., each such node is labelled with a singleton set containing a unique axiom from $\mathcal{D}_{i-1}$). Finally, each of these latter nodes will have an edge to the empty set node l.

If we let $\mathcal{D}_{i-1} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ (that is, $\mathcal{D}_{i-1}$ contains four axioms) then the Lexicographic-tree T for $\mathcal{D}_{i-1}$ is depicted in Figure 4.5.

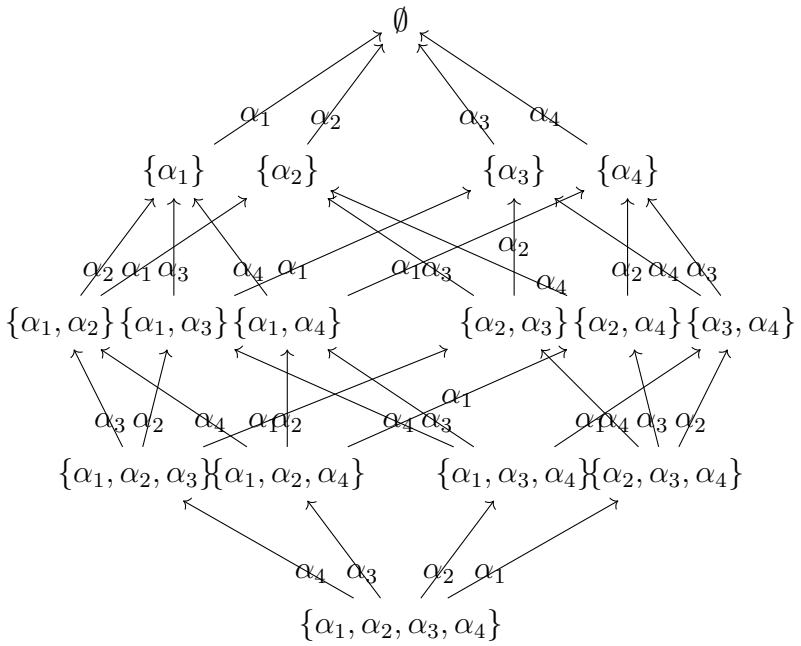


Figure 4.5: Lexicographic-tree for a general problematic rank $\mathcal{D}_{i-1} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$.

Given the definition of a Lexicographic-tree, we can characterise the problems of identifying the LAS and LRAS in terms of tree traversal problems w.r.t. the Lexicographic-tree. Informally, the perspective is that we can traverse the tree (visit each node) starting from the root node and work towards the leaf node. At each point in the traversal (i.e., at each node) we try to add the label of the node (a subset of $\mathcal{D}_{i-1}$) to the maximal C-compatible subset of the ranking. Of course, adding the label of the root node to the maximal C-compatible subset of the ranking will cause C to be exceptional (because $\mathcal{D}_{i-1}$ is the problematic rank). Therefore, we have to continue visiting the children of the root node in a breadth-first fashion. Traversing the tree from the root upwards (i.e., from the largest subsets to the smallest subsets) ensures we will find a maximal subset.

We have depicted the tree in Figure 4.5 in a structured way. I.e., we

have placed the problematic rank (root node) at the bottom of the picture, and all subsets of this rank are placed in "levels" above this node in the picture, in decreasing order of size. Therefore, we can actually assign a number to each level in the picture representing the size of the subsets in that level. Considering this levelled-structure to the tree, starting at the root node begins at level $k = |\mathcal{D}_{i-1}|$, we add the associated knowledge $L_n(r)$ to the ranking and test exceptionality of C. If C is still exceptional we move to level $k-1$ (to the children of the root node whose subsets are of size $k-1$). We then process the labels of each of these children in the same way, one-by-one and in a breadth-first manner.

If at least one of these subsets (when added to the ranking) do not enforce C to be exceptional, then this will be a maximally C-compatible subset of the problematic rank. In other words, if the current level in the tree is k, then we have identified a way of keeping k axioms from the problematic rank. Therefore, we have indirectly identified the LAS because it will constitute the set of all subsets of size k (i.e., those of equal size to the identified maximal C-compatible ones). However, if none of the children represent C-compatible subsets, then we move to a higher level in the tree etc.

Therefore, in essence, the task is to find a minimal path to a node n in the tree s.t. the label of n represents a maximal subset of $\mathcal{D}_{i-1}$ that is compatible with C (when combined with the maximal C-compatible subset of the ranking). Thereafter, the LAS will be the set of all node labels in the tree that have equal size to n. To formalise this, we give a definition for *maximally compatible node* (the node at the end of the minimal path as discussed) and LAS (a revised definition w.r.t. a Lexicographic-tree).

Definition 51 (Maximally Compatible Node and LAS) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB, $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ its ranking, $C \mathrel{\tilde{\sqsubseteq}} D$ a query, $R' = \{\mathcal{D}_n, \ldots, \mathcal{D}_i\}$ the maximal C-compatible subset of R, and let $T = \langle N, E, L_n, L_e \rangle$ be the Lexicographic-tree for $\mathcal{D}_{i-1}$. Then a node $n_1 \in N$ is* maximally compatible *for $\mathcal{D}_{i-1}$ w.r.t. C if $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap \mathcal{C}_{L_n(n_1)} \sqsubseteq \neg C$ and there is no $n_2 \in N$ s.t. $Path(n_2) \subset Path(n_1)$ and $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap \mathcal{C}_{L_n(n_2)} \sqsubseteq \neg C$.*

The LAS *for* T *w.r.t.* R' *and* C *is the set* $\mathcal{D}' = \{L_n(n_2) \mid n_2 \in N$ *and* $|L_n(n_2)| = |L_n(n_1)|\}$.

Definition 51 on Page 219 confirms that a maximally compatible node n in a Lexicographic-tree is (1) a node whose label is compatible with the query antecedent C, and (2) has the shortest path from the root of any node that satisfies (1). It also confirms that the LAS is the set of labels (subsets) in the Lexicographic-tree where each is of equal size to the label of the maximally compatible node(s) in the tree.

Now we have to define a notion of LRAS w.r.t. a Lexicographic-tree. Recall that the LRAS restricts our view to the C-basis (union of all C-justifications) when eliminating axioms from the problematic rank. Therefore, in this case, when traversing the Lexicographic-tree we will only consider paths to nodes n' s.t:

- $\mathcal{D}_{i-1} \backslash \mathscr{C} \subseteq L_n(n')$ (the irrelevant axioms are always retained).
- $Path(n') \subseteq \mathscr{C}$ (we are only interested in nodes that contain subsets of $\mathscr{C}$).

The maximally compatible nodes that we find in this context will be called *maximally-relevant compatible nodes* and the LRAS will be defined in terms of such nodes.

Definition 52 (Maximally-Relevant Compatible Node and LRAS) *Let* $\langle \mathcal{T}, \mathcal{D} \rangle$ *be a LHS-coherent defeasible KB,* $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ *its ranking,* $C \sqsubset\!\!\sim D$ *a query,* $R' = \{\mathcal{D}_n, \ldots, \mathcal{D}_i\}$ *the maximal* C*-compatible subset of* R*,* $T = \langle N, E, L_n, L_e \rangle$ *the Lexicographic-tree for* $\mathcal{D}_{i-1}$*, and* $\mathscr{C}$ *the* C*-basis for* $\langle \mathcal{T}, \mathcal{D} \rangle$*. Then a node* $n_1 \in N$ *is* maximally-relevant compatible *for* $\mathcal{D}_{i-1}$ *w.r.t.* C *if:*

1. $\mathcal{D}_{i-1} \backslash \mathscr{C} \subseteq L_n(n_1)$.
2. $Path(n_1) \subseteq \mathscr{C}$.
3. $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap \mathcal{C}_{L_n(n_1)} \sqsubseteq \neg C$.

4. There is no $n_2 \in N$ *s.t.* $Path(n_2) \subset Path(n_1)$ *and* $\mathcal{T} \not\models \mathcal{C}_{\bigcup R'} \sqcap \mathcal{C}_{L_n(n_2)} \sqsubseteq \neg C$.
The LRAS *for* T *w.r.t.* R' *and* C *is* $\mathcal{D}' = \{L_n(n_3) \mid n_3 \in N \text{ and } |L_n(n_3)| = |L_n(n_1)|\}$.

Finally, we can show the LAS and LRAS correspond by demonstrating that in any Lexicographic-tree there is a maximally compatible node n and maximally-relevant compatible node n' s.t. $n = n'$.

Lemma 19 (LAS and LRAS Correspond) *Let* $\langle \mathcal{T}, \mathcal{D} \rangle$ *be a LHS-coherent defeasible KB,* $R = \{\mathcal{D}_0, \ldots, \mathcal{D}_n\}$ *its ranking,* $C \sqsubsetneq D$ *a query,* $\mathscr{C}$ *the* C*-basis for* $\langle \mathcal{T}, \mathcal{D} \rangle$*, and* $T = \langle N, E, L_n, L_e \rangle$ *the Lexicographic-tree for* $\mathcal{D}_{i-1}$*. Then, there is a maximally compatible node* $n \in N$ *and maximally-relevant compatible node* $n' \in N$ *(w.r.t.* C*) s.t.* $n = n'$.

Proof: We first show an intermediate result: that there is a maximally compatible node $n_1 \in N$ s.t. $Path(n_1) \subseteq \mathscr{C}$. Suppose there is no such node. It implies that for each maximally compatible node $n_1 \in N$, there is an element $x \in Path(n_1)$ s.t. $x \notin \mathscr{C}$. It is straightforward to see that this would contradict with the definition of justification and the definition of maximally compatible node. This is because we can easily choose a node $n_2 \in N$ s.t. $L_n(n_2) = L_n(n_1) \cup \{x \in Path(n_1) \mid x \notin \mathscr{C}\}$. It is clear that $L_n(n_2)$ is compatible with C because we have only added axioms to $L_n(n_1)$ that are irrelevant to the exceptionality of C. This causes a contradiction with n_1 being a *maximally* compatible node because $L_n(n_1) \subset L_n(n_2)$. Therefore, we have proven that there is a maximally compatible node $n_1 \in N$ s.t. $Path(n_1) \subseteq \mathscr{C}$. Now, we have to show that n_1 is also a maximally-relevant node in T w.r.t. C. It is clear that n_1 complies with Requirements 2, 3 and 4 of Definition 52 (by Definition 51 on Page 219). What remains to be shown is that n_1 complies with Requirement 1. I.e., we have to show that $\mathcal{D}_{i-1} \backslash \mathscr{C} \subseteq L_n(n_1)$. This is quite straightforward to see because, by definition of Lexicographic-tree we know that $L_n(n_1) = \mathcal{D}_{i-1} \backslash Path(n_1)$. We have also shown that $Path(n_1) \subseteq \mathscr{C}$ and therefore it immediately follows that $\mathcal{D}_{i-1} \backslash \mathscr{C} \subseteq L_n(n_1)$. Therefore, n_1 is

a maximally-relevant compatible node for T w.r.t. C. □

Lemma 19 on Page 221 clearly shows that the LAS and LRAS will be equal for any Lexicographic-tree because the maximally compatible and maximally-relevant compatible nodes are the same. Adding the internalisation mechanism back to the fold it follows immediately that the LAC and LRAC also correspond. Finally, this result proves that Lexicographically-Relevant Closure and Lexicographic Closure correspond. This means that Lexicographically-Relevant Closure is a potentially optimised way in which to compute Lexicographic Closure.

It must be mentioned that we have presented just three instances of Relevant Closure. Indeed these are the only investigated notions of Relevant Closure at present. As we have stated earlier, Relevant Closure defines numerous inference procedures because, based on the relevance notion of justifications, there are numerous ways to use axioms in justifications to "repair" the exceptionality of the entailment. In other words, for Basic, Minimal and Lexicographically-Relevant Closures we have employed just three possible ways. In theory, there are other "repair" approaches in this class to investigate. Some may have more logical merit than others. Nevertheless, knowledge of these gaps in Relevant Closure research remains impetus for future work.

4.6 Optimisations

In this fairly brief section we demonstrate some results which suggest useful optimisations for the procedures discussed in Sections 4.3 to 4.5. The first result is an interesting relationship between a defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and its classical counterpart $\mathcal{T} \cup \mathcal{D}'$. The result was actually shown in the left to right direction of the "proof" for Proposition 1 on Page 137. However, we give a more explicit and intuitive representation here for this context.

Lemma 20 (Exceptionality Implies Classical Unsatisfiability) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB and $\mathcal{T} \cup \mathcal{D}'$ its classical counterpart. If C is exceptional w.r.t. $\langle \mathcal{T}, \mathcal{D} \rangle$, then C is unsatisfiable w.r.t. $\mathcal{T} \cup \mathcal{D}'$ for any concept C.*

Modern DL reasoners are highly optimised to check satisfiability or unsatisfiability of concepts [182, 191, 90]. Therefore, if we want to verify if a concept is exceptional or not, we can, as an approximation first check if it is unsatisfiable w.r.t. the classical counterpart of the KB (because it can only be exceptional if it is unsatisfiable). This would almost certainly save many exceptionality checks on satisfiable concepts seeing as, in practice, the number of satisfiable concepts vastly outweigh the number of unsatisfiable concepts in real-world ontologies. Therefore, to identify all the exceptional concepts in the KB, we can first identify all the unsatisfiable concepts first to vastly narrow the search space, and then recurse on this much smaller set to identify the exceptional concepts using the reduction mentioned in Theorem 1 on Page 143. This optimisation is applicable for the computation of the ranking and for deciding entailment using a preferential reasoning paradigm.

On a related note, we saw in Lemma 9 on Page 165 that the ranking of a defeasible KB respects specificity. This result leads to another possible optimisation for computation of the ranking which informally states: "all subclasses of exceptional things are exceptional". More formally:

Lemma 21 (Exceptionality Propagates Through Subsumption) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB and C_1 a concept s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \mathrel{\dot{\sqsubseteq}} \neg C_1$. Then, for any concept C_2 s.t. $\mathcal{T} \models C_2 \sqsubseteq C_1$, $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \mathrel{\dot{\sqsubseteq}} \neg C_2$.*

Proof: Suppose that $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \mathrel{\dot{\sqsubseteq}} \neg C_1$ and there is a concept C_2 s.t. $\mathcal{T} \models C_2 \sqsubseteq C_1$ but that $\langle \mathcal{T}, \mathcal{D} \rangle \not\models_r \top \mathrel{\dot{\sqsubseteq}} \neg C_2$. Therefore, there is a ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. there is an $x \in {C_2}^{\mathcal{R}}$ and $rk_{\mathcal{R}}(x) = 0$. Let $\mathcal{I}' = \langle \Delta^{\mathcal{I}'}, \cdot^{\mathcal{I}'} \rangle$ be a classical interpretation s.t. $\Delta^{\mathcal{I}'} = \Delta^{\mathcal{R}}$ and $\cdot^{\mathcal{I}'} = \cdot^{\mathcal{R}}$. We know that $\mathcal{I}' \Vdash \mathcal{T}$ because $\mathcal{R} \Vdash \mathcal{T}$, $\Delta^{\mathcal{I}'} = \Delta^{\mathcal{R}}$ and $\cdot^{\mathcal{I}'} = \cdot^{\mathcal{R}}$. But from our assumption that $\mathcal{T} \models C_2 \sqsubseteq C_1$ it follows that $\mathcal{I}' \Vdash C_2 \sqsubseteq C_1$. This means that

$x^{\mathcal{I}'} \in {C_2}^{\mathcal{I}'} \cap {C_1}^{\mathcal{I}'}$. But this must mean that $x^{\mathcal{R}} \in {C_1}^{\mathcal{R}}$ because $\Delta^{\mathcal{I}'} = \Delta^{\mathcal{R}}$ and $\cdot^{\mathcal{I}'} = \cdot^{\mathcal{R}}$. This means that $\mathcal{R} \not\Vdash \top \,\underset{\sim}{\sqsubseteq}\, \neg C_1$ which is a contradiction with $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \,\underset{\sim}{\sqsubseteq}\, \neg C_1$. Therefore, $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \,\underset{\sim}{\sqsubseteq}\, \neg C_2$. □

The result of Lemma 21 on Page 223 allows us to infer that a concept is exceptional if we know it is more specific than a known exceptional concept (using syntactic, heuristic or other techniques). This optimisation can be used in numerous applications of defeasible reasoning. It should be particularly useful when updating the ranking of a KB after it is incrementally modified (axioms are added or removed). This would allow one to avoid recomputing the entire ranking from scratch after the KB is modified.

In fact, Lemma 21 on Page 223 hints at another useful result for incremental update of the ranking. Loosely speaking, it identifies that "subsets of exceptional things are at least as exceptional as these things". That is, the following lemma concerning the ranks of more specific concepts holds.

Lemma 22 (Degree of Exceptionality Respects Subsumption) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a LHS-coherent defeasible KB and C_1 a concept s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \models_r \top \,\underset{\sim}{\sqsubseteq}\, \neg C_1$. Then, for any concept C_2 s.t. $\mathcal{T} \models C_2 \sqsubseteq C_1$, $rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(C_2) \geq rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(C_1)$.*

Proof: Suppose that $rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(C_2) < rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(C_1)$. By definition of $rk_{\langle \mathcal{T}, \mathcal{D} \rangle}$ we know there is a ranked model $\mathcal{R}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. $rk_{\mathcal{R}}(C_2) = i$ and there is no other ranked model for $\langle \mathcal{T}, \mathcal{D} \rangle$ where the rank of C_2 is lower than i. The same can be said of C_1. We pick this relevant ranked model $\mathcal{R}'$ for C_1. We pick an $x \in \min_{\prec_{\mathcal{R}}}({C_2}^{\mathcal{R}})$. From $\mathcal{T} \models C_2 \sqsubseteq C_1$ we know that $x \in {C_1}^{\mathcal{R}}$. But this means that $rk_{\mathcal{R}}(C_2) \geq rk_{\mathcal{R}'}(C_1)$ which is a contradiction with $rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(C_2) < rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(C_1)$. Therefore, $rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(C_2) \geq rk_{\langle \mathcal{T}, \mathcal{D} \rangle}(C_1)$. □

Lemma 22 on Page 224 says that concepts that are more specific than some exceptional concept are at least as exceptional as this concept. It is a very straightforward result but very useful to avoid naïve recomputation of the ranking when the KB is incrementally modified.

Recall that in Section 4.5, the presented Relevant Closures require computation of justifications in order to avoid removing irrelevant or unnecessary axioms to the exceptionality in question. This is certainly one way to avoid unnecessary entailment checks. However, justification finding is a computationally complex task and standard DL reasoning tasks need to be used (or modified in the case of glassbox approaches) to pinpoint them. There is another approach to prune away irrelevant axioms to the exceptionality that does not require DL reasoning. This approach uses the principle of *module* for a KB w.r.t. a signature (set of terms from the KB) [83, 170]. Intuitively, a module of a KB w.r.t. a signature is a small subset of the ontology which preserves the meaning of the terms in the signature.

The topic of modularisation of ontologies is a broad and involved research area. For our purposes, we are interested in applying the techniques of modularisation to prune away irrelevant axioms in the ranking (without using reasoning). So for our context, the signature of interest would be the set of terms used in our reasoning query. Intuitively, what we want to do is extract a module of our KB that eliminates as many irrelevant axioms as possible, but at the same time not eliminating any axioms that contribute to the exceptionality of the query. Therefore we are interested in modules which have two properties: (1) they should preserve all justifications for all entailments that can be formed over the signature of our query. I.e., the module should retain all axioms relevant to our query. And (2) the module should be as small as possible (the module should exclude as many axioms as possible to reduce entailment checks and increase the performance).

So-called *star locality-based modules* [170] satisfy both of these properties. However, since we are in the context of defeasible reasoning, and modularisation is predominantly developed for classical reasoning systems, one may ask if the property of preserving all justifications for all entailments over the given signature actually translates to the defeasible case. The answer is yes because we have shown extensively that defeasible reasoning (in $\mathcal{ALC}$) can be reduced to classical DL entailment.

4.7 Syntactic Sugar

We recall from Section 2 when we introduced DLs, that one can express useful abbreviations (concise syntactic forms) for subsumption statements. We call such abbreviations "syntactic sugar" because they do not add to the expressivity of DLs but just offer more concise (and sometimes more intuitive) representations of certain semantic relationships. The main ones we have discussed for classical DLs are *equivalence* and *disjointness* statements.

For example, the equivalence statement $\mathsf{Man} \equiv \mathsf{Person} \sqcap \mathsf{Male}$ (men and persons who are male refer to the same objects in the domain) is an abbreviation for the two subsumptions $\mathsf{Man} \sqsubseteq \mathsf{Person} \sqcap \mathsf{Male}$ and $\mathsf{Person} \sqcap \mathsf{Male} \sqsubseteq \mathsf{Man}$. That is to say, the two subsumptions and the equivalence statement have the same models (i.e., they are satisfied in the same interpretations).

Similarly, the disjointness statement $\mathsf{Man} \parallel \mathsf{Woman}$ (the set of objects in the domain referred to as men, and the set referred to as woman, are disjoint) is an abbreviation for the subsumption $\mathsf{Man} \sqcap \mathsf{Woman} \sqsubseteq \bot$. Of course, these correspondences are a convenient result of the semantics that DLs have assigned to the constructs of subsumption, equivalence and disjointness.

The questions now are: is it useful and intuitive to be able to talk about *defeasible* versions of equivalence and disjointness and, if so, what semantics should we assign to such notions in the preferential context? To address the first question we present some examples to make it evident that humans often *do* make defeasible equivalence and defeasible disjointness statements.

4.7.1 Defeasible Equivalence

In the area of advertising, many companies and business analysts use the terms *digital marketing* and *online marketing* interchangeably. In other words, they usually refer to the same principle. Considering that digital marketing refers to advertising through digital media channels, it is clear that online marketing (for example through social media) is a form of digital marketing. On the other hand, it may be contentious to state that all digital

marketing takes place through the Web.

Television and Radio advertising is usually propagated through digital media and, although it is sometimes transmitted online as well, the traditional "offline" channels remain the ones reaching the majority of customers. It therefore makes sense to say that the concepts DigitialMarketingCampaign (DMC for short) and OnlineMarketingCampaign (OMC for short) are *usually* equivalent.

Now, in the framework of ranked model semantics, what semantics can one assign to such a statement? We recall that we interpreted a defeasible subsumption $C \sqsubseteq D$ as "typical C's are D's". Seeing as classical equivalence corresponds to the rendering of subsumption *in both directions* for the given concepts, we experiment with such an approach in the defeasible context to study its semantic intuition.

Reversing the direction of this subsumption, we arrive at "typical D's are C's" (or the subsumption $D \sqsubseteq C$). Advocating both these subsumptions to define defeasible equivalence seems to give back an intuitive semantic representation for defeasible equivalence. Essentially these subsumptions together restrict our view to the models in which the typical C's and typical D's coincide.

That is, where the typical C's are typical D's are classically equivalent. Of course, we can only study the merit of this semantic definition by examining how it relates to our intuition in real-world examples. In the digital vs. online marketing example, it certainly does make sense to talk about typical digital marketing campaigns and typical online marketing campaigns coinciding. However, only one direction of this defeasible subsumption relationship is contentious from an intuitive perspective.

That is, it is clear that online marketing falls within the jurisdiction of digital marketing, but the converse constraint may have exceptions. In other words, we should only enforce that *typical* digital marketing campaigns are online marketing campaigns. Formally speaking, we agree upon the following definition for defeasible equivalence:

Definition 53 (Defeasible Equivalence) *Two concepts C and D are defeasibly equivalent (written as $C \approx D$) in a ranked interpretation $\mathcal{R}$ if:* $\min_{\prec_\mathcal{R}}(C^\mathcal{R}) = \min_{\prec_\mathcal{R}}(D^\mathcal{R})$.

Besides its intuitive merit, this representation borrows a useful correspondence from the classical case. That is, the semantic relationship between defeasible equivalence and defeasible subsumption is analogous to the relationship between classical equivalence and classical subsumption.

In other words, given the semantics we have assigned, defeasible equivalence can be represented in terms of defeasible subsumption. Therefore, for our marketing example the statement $\mathsf{DMC} \approx \mathsf{OMC}$ is logically equivalent to $\mathsf{DMC} \mathrel{\tilde{\sqsubset}} \mathsf{OMC}$, $\mathsf{OMC} \mathrel{\tilde{\sqsubset}} \mathsf{DMC}$ collectively.

The proof demonstrating the reduction of defeasible equivalence to defeasible subsumption is straightforward and we therefore do not specify it here.

4.7.2 Defeasible Disjointness

Recall from Section 2.1 that one can express that two concepts C and D are disjoint (they cannot share any elements), and we denote this by $C \parallel D$. For example, we know that birds and plants are disjoint ($\mathsf{Bird} \parallel \mathsf{Plant}$), married men are disjoint with bachelors ($\mathsf{Man} \sqcap \exists \mathsf{marriedTo}.\top \parallel \mathsf{Bachelor}$) etc. Is there a case for *defeasible* disjointness?

We feel that there is, and we give an interesting example as a case in point. In describing human physiology, it may make sense to add a constraint to our knowledge that people whose eye irises are blue are disjoint with those whose eye irises are green. However, there are exceptions to this constraint and one example of this is expressed through the condition known as *Heterochromia*[3].

The condition defines cases in which people have distinct colours either in one eye iris, or in the irises of each eye. In light of this condition, and other conditions like it, it seems useful to be able to represent that two concepts

[3] `nlm.nih.gov/medlineplus/ency/article/003319.htm`

are *usually* disjoint, thereby avoiding logical incoherence and inconsistencies caused by exceptions.

Whereas the typical C's and D's *coinciding* seemed an appropriate definition for defeasible equivalence, for defeasible disjointness the typical C's and D's being *distinct* seems a compelling definition. That is, the typical C's and D's are disjoint:

Definition 54 (Defeasible Disjointness) *Two concepts C and D are defeasibly disjoint (written as $C \wr\wr D$) in a ranked interpretation $\mathcal{R}$ if:* $\min_{\prec_\mathcal{R}}(C^\mathcal{R}) \cap \min_{\prec_\mathcal{R}}(D^\mathcal{R}) = \emptyset$.

For our Heterochromia example, Definition 54 on Page 229 essentially holds that: blue-eyed people are usually disjoint with green-eyed people, if and only if, *typical* people with blue eyes, and *typical* people with green eyes, are distinct.

The question now arises if defeasible disjointness is actually syntactic sugar or do we need to enrich the expressivity of defeasible DLs to be able to capture this principle? A study of the semantics given in Definition 54 on Page 229 as well as ranked model semantics gives us a clue as to a representation of defeasible disjointness in terms of defeasible subsumption.

That is, we arrive at the following defeasible subsumption rendering: *typical* blue-eyed people and *typical* green-eyed people will be disjoint, if and only if, the most typical objects in our domain that are either blue-eyed or green-eyed are not *both* blue and green-eyed. This reduction is quite intuitive and lends more credibility to the semantics of Definition 54 on Page 229.

We give a proof for the reduction of defeasible disjointness to defeasible subsumption here because it is perhaps not as straightforward as it is for defeasible equivalence.

Lemma 23 (Defeasible Disjointness w.r.t. Defeasible Subsumption) *For any ranked interpretation $\mathcal{R}$, $\mathcal{R} \Vdash C \wr\wr D$ if and only if $\mathcal{R} \Vdash C \sqcup D \sqsubseteq \neg(C \sqcap D)$.*

Proof: $\Longrightarrow$ we pick a ranked interpretation R s.t. $\min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \cap \min_{\prec_{\mathcal{R}}}(D^{\mathcal{R}}) = \emptyset$. We assume that $\mathcal{R} \not\Vdash C \sqcup D \,\tilde{\sqsubseteq}\, \neg(C \sqcap D)$ and try to derive a contradiction. We know there is an $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}} \cup D^{\mathcal{R}})$ s.t. $x \in C^{\mathcal{R}} \cap D^{\mathcal{R}}$. This means that there is no $y \prec_{\mathcal{R}} x$ s.t. $y \in C^{\mathcal{R}}$ and similarly there is no $z \prec_{\mathcal{R}} x$ s.t. $z \in D^{\mathcal{R}}$. This clearly implies that $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}})$ *and* $x \in \min_{\prec_{\mathcal{R}}}(D^{\mathcal{R}})$. I.e., $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \cap \min_{\prec_{\mathcal{R}}}(D^{\mathcal{R}})$. This is a contradiction with our assumption that $\min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \cap \min_{\prec_{\mathcal{R}}}(D^{\mathcal{R}}) = \emptyset$. □

$\Longleftarrow$ we pick a ranked interpretation s.t. $\mathcal{R} \Vdash C \sqcup D \,\tilde{\sqsubseteq}\, \neg(C \sqcap D)$ and we assume that $\min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \cap \min_{\prec_{\mathcal{R}}}(D^{\mathcal{R}}) \neq \emptyset$. We try to derive a contradiction from this. From the latter assumption we know that there is an $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \cap \min_{\prec_{\mathcal{R}}}(D^{\mathcal{R}})$. Therefore, there can be no $y \prec_{\mathcal{R}} x$ s.t. either $y \in C^{\mathcal{R}}$ or $y \in D^{\mathcal{R}}$. This advertently means that $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}} \cup D^{\mathcal{R}})$ which, given our selected $\mathcal{R}$, implies that $x \notin C^{\mathcal{R}} \cap D^{\mathcal{R}}$. This is clearly a contradiction with the fact that $x \in \min_{\prec_{\mathcal{R}}}(C^{\mathcal{R}}) \cap \min_{\prec_{\mathcal{R}}}(D^{\mathcal{R}})$. □

Finally, we conclude this section by pointing out that our notion of defeasible disjointness, while a reasonable characterisation, is possibly not the only reasonable notion one can devise. There may be other sensible notions within the preferential framework, and we ourselves have considered two possible definitions borrowed from classical disjointness.

In classical disjointness we know that the axiom $C \parallel D$ can be equivalently represented by the subsumptions $C \sqsubseteq \neg D$ or $C \sqcap D \sqsubseteq \bot$.

In principle, the defeasible counterparts of these statements can be used to expression a notion of defeasible disjointness. However, the defeasible counterpart of the latter statement would not be suitable to capture defeasible disjointness.

That is, the ranked interpretations satisfying the statement $C \sqcap D \,\tilde{\sqsubseteq}\, \bot$ are the same as those satisfying $C \sqcap D \sqsubseteq \bot$ (classical disjointness).

The other possibility, namely $C \,\tilde{\sqsubseteq}\, \neg D$, is more interesting. It says that typical C's should not be D's. Because typicality is restricted only to the LHS with defeasible subsumption, this notion does not treat C and D sym-

metrically.

That is, the statements $C \not\sqsubseteq \neg D$ and $D \not\sqsubseteq \neg C$ are not logically equivalent (whereas their classical counterparts are). Thus a sensible notion of defeasible disjointness needs to take into account both these defeasible subsumptions in some way. Nevertheless, we leave such an investigation as future work.

4.8 Discussion

In this chapter we have presented how the principle of concept exceptionality can be reduced to classical entailment (Section 4.1). This base result paves the way to be able to characterise and compute the core structure of all our preferential reasoning paradigms - the exceptionality ranking of a defeasible KB (Section 4.2). The subsequent algorithmic constructions we gave (Sections 4.3 to 4.5) essentially varied only in the notion of maximal C-compatibility of the ranking that they subscribe to.

In the case of Lexicographic Closure 4.4 we also gave a semantic characterisation in terms of ranked models which corresponds to our presented procedure. We showed that Lexicographic Closure is an inferential extension of Rational Closure, while Minimal Relevant Closure is an inferential extension of Basic Relevant Closure (though the latter two procedures are non-rational in the KLM sense). We also defined a type of Relevant Closure called the Lexicographically-Relevant Closure which corresponds exactly with Lexicographic Closure, and defines an optimised procedure for computing the latter. In the next chapter we show which requirements (argued for in Chapter 3) are satisfied by each of the constructions in this chapter. We also perform similar evaluations for some of the non-preferential proposals mentioned in Chapter 2.

Finally, as a matter of philosophical interest, since the procedures we have presented are meant to address monotonicity of classical entailment, they must be nonmonotonic in nature. However, we have shown that these procedures actually reduce to classical entailment. Some readers may in-

quire at this point how to intuitively understand this. One could consider the following explanation: suppose I am an agent who cannot comprehend defeasible statements. That is, I can only reason about strict statements (like our presented reasoning procedures). Then I can still adhere to a defeasible reasoning paradigm when reconciling my knowledge, using the principle of *selective ignorance*.

In other words, if I accept as fact that students don't pay taxes (strict information), but I later encounter a specific student (for example an employed one) who does pay taxes, then I have a choice: I can either choose to ignore this latest finding so that I can still accept my previous finding. Or, I can choose to ignore the previous finding to accept the latter one. Of course, the presented procedures in this chapter adhere to the latter paradigm (more specific knowledge is more important).

Chapter 5

Evaluating the Inferences of Defeasible Reasoning

In Chapter 3 we made a case for the satisfaction of the KLM postulates by arguing that a defeasible reasoning mechanism, in the context of DLs, should at the very least satisfy these properties in order to guarantee sensible reasoning behaviour. This chapter evaluates the preferential reasoning algorithms presented in Chapter 4, and the applicable alternatives in Chapter 2, against these formal properties motivated in Chapter 3.

We first evaluate our preferential algorithms against the formal KLM properties given in Section 3.2.1 of Chapter 3. Thereafter, we consider the alternative defeasible reasoning approaches mentioned in Chapter 2 against the same requirements. We conclude with a short discussion about the significance of the results.

5.1 Preferential Algorithms

In Chapter 2 we have presented overviews of various defeasible reasoning formalisms that address the exception problem in DLs. Among those presented proposals, we chose to further develop the preferential approach towards the pragmatic goals of this thesis.

The main reasons we chose the preferential approach were 1) its generality as a framework for defeasible reasoning and 2) the structural properties that it respects. These characteristics make the inferential behaviours of entailment regimes built on top of the framework, have a more clearly defined "shape" or "structure".

The framework as a whole is also arguably more conceptually simple and elegant than other formalisms, in part because it handles the matter of priorities internally without need of user input. Notwithstanding, restricted refinement of priorities is still permitted in the framework (see Section 4.2).

In this section we evaluate the preferential reasoning algorithms (presented in Chapter 4) against the advocated KLM properties motivated in Section 3.2.1. Since preferential DLs introduce a new notion of subsumption ($\sqsubseteq$) in the language, and it has a clear ranked model semantics, it is very easy to substitute this notion for $\rightsquigarrow$ defined in Section 3.2.1 and evaluate it against the postulates.

Examining current literature concerning the preferential approach (as applied to DLs) it is surprising to note that there is no *explicit* demonstration that $\sqsubseteq$, when interpreted in any ranked interpretation, actually satisfies all the KLM postulates. This is of course the object-level perspective of the relation. This result is expressed in the following lemma, the proof of which is fairly straightforward:

Lemma 24 (Defeasible Subsumption Satisfies KLM Postulates) *Let $\mathcal{R}$ be a ranked interpretation. Then, for any three concepts C, D and E:*

1. $\mathcal{R} \Vdash C \sqsubseteq C$ *(Ref)*
2. *If* $\mathcal{R} \Vdash C \equiv D$ *and* $\mathcal{R} \Vdash C \sqsubseteq E$ *then* $\mathcal{R} \Vdash D \sqsubseteq E$ *(LLE)*
3. *If* $\mathcal{R} \Vdash C \sqsubseteq D$ *and* $\mathcal{R} \Vdash C \sqsubseteq E$ *then* $\mathcal{R} \Vdash C \sqsubseteq D \sqcap E$ *(And)*
4. *If* $\mathcal{R} \Vdash C \sqsubseteq D$ *and* $\mathcal{R} \Vdash E \sqsubseteq D$ *then* $\mathcal{R} \Vdash C \sqcup E \sqsubseteq D$ *(Or)*
5. *If* $\mathcal{R} \Vdash C \sqsubseteq D$ *and* $\mathcal{R} \Vdash D \sqsubseteq E$ *then* $\mathcal{R} \Vdash C \sqsubseteq E$ *(RW)*

6. If $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} D$ *and* $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} E$ *then* $\mathcal{R} \Vdash C \sqcap D \mathrel{\sqsubseteq\!\!\!\sim} E$ *(CM)*

7. If $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} D$ *and* $\mathcal{R} \nVdash C \mathrel{\sqsubseteq\!\!\!\sim} \neg E$ *then* $\mathcal{R} \Vdash C \sqcap E \mathrel{\sqsubseteq\!\!\!\sim} D$ *(RM)*

Proof: 1. $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} C$ if and only if $\min_{\prec_\mathcal{R}}(C^\mathcal{R}) \subseteq C^\mathcal{R}$ and the latter follows by definition of the function $\min_{\prec_\mathcal{R}}$ (see Definition 7 on Page 85).
2. (LLE) follows easily as well: If $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} E$ (the minimal C's are all E's) and $\mathcal{R} \Vdash C \equiv D$ (C and D correspond to the same set of elements) we can deduce that the minimal C's are equivalent to the minimal D's and hence it follows that $\mathcal{R} \Vdash D \mathrel{\sqsubseteq\!\!\!\sim} E$ (the minimal C's are D's).
3. (And) is straightforward: If $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} D$ (the minimal C's are all D's) and $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} E$ (the minimal C's are also all E's) then $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} D \sqcap E$ (the minimal C's are in the intersection of D and E). In other words, if $\min_{\prec_\mathcal{R}}(C^\mathcal{R}) \subseteq D^\mathcal{R}$ *and* $\min_{\prec_\mathcal{R}}(C^\mathcal{R}) \subseteq E^\mathcal{R}$ then $\min_{\prec_\mathcal{R}}(C^\mathcal{R}) \subseteq D^\mathcal{R} \cap E^\mathcal{R}$.
4. Assume that $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} D$ and $\mathcal{R} \Vdash E \mathrel{\sqsubseteq\!\!\!\sim} D$ but that $\mathcal{R} \nVdash C \sqcup E \mathrel{\sqsubseteq\!\!\!\sim} D$. There is an $x \in \min_{\prec_\mathcal{R}}((C \sqcup E)^\mathcal{R})$ s.t. $x \notin D^\mathcal{R}$. There are three cases:
Case 1*:* $x \in C^\mathcal{R}$ and $x \notin E^\mathcal{R}$. By definition of $\min_{\prec_\mathcal{R}}$, there is no $y \prec_\mathcal{R} x$ s.t. $y \in (C \sqcup E)^\mathcal{R}$. That is, for each $y \prec_\mathcal{R} x$, $y \notin C^\mathcal{R}$ and $y \notin E)^\mathcal{R}$. This means that $x \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$. But from our assumption that $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} D$ it means $x \in D^\mathcal{R}$. This is a contradiction with our assumption that $x \notin D^\mathcal{R}$.
Case 2*:* $x \in E^\mathcal{R}$ and $x \notin C^\mathcal{R}$ (Symmetric to Case 1).
Case 3*:* $x \in C^\mathcal{R}$ and $x \in E^\mathcal{R}$. By definition of $\min_{\prec_\mathcal{R}}$, there is no $y \prec_\mathcal{R} x$ s.t. $y \in (C \sqcup E)^\mathcal{R}$. That is, for each $y \prec_\mathcal{R} x$, $y \notin C^\mathcal{R}$ and $y \notin E)^\mathcal{R}$. Therefore, $x \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$ and $x \in \min_{\prec_\mathcal{R}}(E^\mathcal{R})$. From both our assumptions $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} D$ and $\mathcal{R} \Vdash E \mathrel{\sqsubseteq\!\!\!\sim} D$ we derive that $x \in D^\mathcal{R}$. Again, we arrive at a contradiction with our assumption that $x \notin D^\mathcal{R}$.
5. $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} D$ implies that $\min_{\prec_\mathcal{R}}(C^\mathcal{R}) \subseteq D^\mathcal{R}$. $\mathcal{R} \Vdash D \sqsubseteq E$ implies that $D^\mathcal{R} \subseteq E^\mathcal{R}$. By transitivity of $\subseteq$ (subset inclusion), we obtain that $\min_{\prec_\mathcal{R}}(C^\mathcal{R}) \subseteq E^\mathcal{R}$. I.e., that $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} E$.
6. Assume that $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} D$ and $\mathcal{R} \Vdash C \mathrel{\sqsubseteq\!\!\!\sim} E$ but that $\mathcal{R} \nVdash C \sqcap D \mathrel{\sqsubseteq\!\!\!\sim} E$. There is an $x \in \min_{\prec_\mathcal{R}}((C \sqcap D)^\mathcal{R})$ s.t. $x \notin E^\mathcal{R}$. We know obviously from $x \in \min_{\prec_\mathcal{R}}((C \sqcap D)^\mathcal{R})$ that $x \in (C \sqcap D)^\mathcal{R})$. There are two cases:

Case 1*:* $x \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$. From $\mathcal{R} \Vdash C \sqsubseteq\!\!\!\sim E$ we know that $x \in E^\mathcal{R}$. This is a contradiction with our assumption that $x \notin E^\mathcal{R}$.
Case 2*:* $x \notin \min_{\prec_\mathcal{R}}(C^\mathcal{R})$. There is a $y \prec_\mathcal{R} x$ s.t. $y \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$. We pick such a y. Therefore, from $\mathcal{R} \Vdash C \sqsubseteq\!\!\!\sim D$ we know that $y \in D^\mathcal{R}$. But this contradicts with $x \in \min_{\prec_\mathcal{R}}((C \sqcap D)^\mathcal{R})$ which says that there is no $z \prec_\mathcal{R} x$ s.t. $z \in C^\mathcal{R} \cap D^\mathcal{R}$.
7. Assume that $\mathcal{R} \Vdash C \sqsubseteq\!\!\!\sim D$ and $\mathcal{R} \nVdash C \sqsubseteq\!\!\!\sim \neg E$ but that $\mathcal{R} \nVdash C \sqcap E \sqsubseteq\!\!\!\sim D$. Our second assumption implies that there is an $x \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$ s.t. $x \in E^\mathcal{R}$. From our third assumption, there is a $y \in \min_{\prec_\mathcal{R}}((C \sqcap E)^\mathcal{R})$ s.t. $y \notin D^\mathcal{R}$. We have three cases:
Case 1*:* $x \prec_\mathcal{R} y$. From $\mathcal{R} \nVdash C \sqsubseteq\!\!\!\sim \neg E$ we know that $x \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$ and $x \in E^\mathcal{R}$. This contradicts with $y \in \min_{\prec_\mathcal{R}}((C \sqcap E)^\mathcal{R})$ (i.e., that there is no element more minimal than y also belonging to both C and E).
Case 2*:* $y \prec_\mathcal{R} x$. Contradicts with $x \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$ because $y \in C^\mathcal{R}$ (from $y \in \min_{\prec_\mathcal{R}}((C \sqcap E)^\mathcal{R})$).
Case 3*:* $rk_\mathcal{R}(x) = rk_\mathcal{R}(y)$. From $\mathcal{R} \Vdash C \sqsubseteq\!\!\!\sim D$ we know that $x \in D^\mathcal{R}$. But if x is of equal rank to y and x is contained in $\min_{\prec_\mathcal{R}}(C^\mathcal{R})$ then it must be the case that $y \in \min_{\prec_\mathcal{R}}(C^\mathcal{R})$. From our assumption that $\mathcal{R} \Vdash C \sqsubseteq\!\!\!\sim D$ it means $y \in D^\mathcal{R}$. This contradicts with our earlier assumption that $y \notin D^\mathcal{R}$. □

Lemma 24 on Page 234 shows that, restricting ourselves to a single ranked interpretation, the semantics of defeasible subsumption satisfies all the KLM postulates.

Unfortunately, the literature also does not demonstrate definitively which defeasible entailment regimes (induced by defeasible subsumption) satisfy which postulates. I.e., to the best of our knowledge, the proofs for this meta-level consideration of the KLM postulates are not explicitly provided.

Even though the majority of these proofs are fairly straightforward, we provide them here for completeness and ease of reference. It must be emphasised though that the results for Rational and Lexicographic Closure are accepted and known by the preferential reasoning community in general.

We start off by showing that both Rational Closure and Lexicographic Closure induce rational consequence relations. In other words, they each satisfy all the formal KLM postulates.

It turns out that we do not need involved proofs to show this. In fact this result follows from Lemma 24 and the fact that both Rational and Lexicographic Closure can be characterised in terms of single ranked models (see Definitions 23 and 38 on Pages 97 and 185 respectively). Furthermore, we have shown Lexicographically-Relevant Closure to be an equivalent construction to Lexicographic Closure (see Lemma 19 on Page 221). This means that Lexicographically-Relevant Closure also computes a rational consequence relation.

Within the preferential reasoning algorithms, this leaves us with the Basic and Minimal Relevant Closures. We have stated in Section 4.5 that these constructions do not define rational consequence relations. I.e., they do not satisfy all the KLM postulates. Here we demonstrate which of the postulates they *do* in fact satisfy, as well as provide applicable counter-examples to show which of the postulates they do *not* satisfy.

Theorem 2 (The Relevant Closures w.r.t. KLM Postulates) *The Basic and Minimal Relevant Closures satisfy the properties (Ref), (And), (RW) and (LLE) and do not satisfy the properties (Or), (CM) and (RM).*

Proof: we show that for any defeasible KB $\mathcal{K}$: 1. (Ref) - for any concept C, $C \sqsubseteq C$ will be in the basic and minimal Relevant Closures of $\mathcal{K}$. By Definition 47 on Page 205, this is the case if and only if $C \sqsubseteq C$ is classically entailed by the classical counterpart of the maximally-relevant C-compatible subset of the ranking for $\mathcal{K}$ (both notions of maximal relevance apply here for each closure). In other words, our procedures reduce to classical subsumption checking. By the semantics of classical subsumption, we know that any set of elements is a non-strict subset of itself, therefore both closures satisfy (Ref).

2. (And) - if $C \sqsubseteq D$ and $C \sqsubseteq E$ is in the a basic (resp. minimal) Relevant

Closure for $\mathcal{K}$, then $C \sqsubseteq\!\!\!\!\sim D \sqcap E$ is also in the basic (resp. minimal) Relevant Closure for $\mathcal{K}$. But the premises of this argument imply that the maximally-relevant C-compatible subset of the ranking is s.t. $C \sqsubseteq D$ and $C \sqsubseteq E$ are both classically entailed by its classical counterpart. Because the (And) rule is satisfied by classical entailment we obtain that $C \sqsubseteq D \sqcap E$ is also classically entailed by its classical counterpart. Therefore, both closures also satisfy the (And) property.

3. (LLE) - if $C \sqsubseteq\!\!\!\!\sim E$ and $C \equiv D$ are in the basic (resp. minimal) Relevant Closure for $\mathcal{K}$, then $D \sqsubseteq\!\!\!\!\sim E$ is also in the basic (resp. minimal) Relevant Closure for $\mathcal{K}$. The first premise means that $C \sqsubseteq E$ classically follows from the maximally-relevant C-compatible subset of the ranking for $\mathcal{K}$. The second premise can be understood as follows: all our preferential reasoning algorithms imply that any strict query such as $C \equiv D$ will be in the selected defeasible closure of the KB if and only if it follows classically from the TBox (strict information) in the KB alone. By monotonicity of classical entailment we would also have that $C \equiv D$ follows classically when adding the classical counterpart of the maximally-relevant C-compatible subset of the ranking to the TBox. Thus, from both our derivations $C \equiv D$ and $C \sqsubseteq E$, as well as classical reasoning, we can easily infer $D \sqsubseteq E$. Therefore, both closures will satisfy (LLE).

4. (RW) - if $C \sqsubseteq\!\!\!\!\sim D$ and $D \sqsubseteq E$ are in the basic (resp. minimal) Relevant Closure for $\mathcal{K}$, then $C \sqsubseteq\!\!\!\!\sim E$ is also in the basic (resp. minimal) Relevant Closure for $\mathcal{K}$. A similar argument to (LLE) is used. $D \sqsubseteq E$ can be derived classically from the strict information alone. From the union of this strict information and the classical counterpart of the maximally-relevant C-compatible subset of the ranking we can classically derive $C \sqsubseteq D$. By monotonicity of classical entailment we can derive both the above conclusions from this latter union of sets. Using classical reasoning it is easy to see we can derive $C \sqsubseteq E$ via transitivity of classical subsumption. Therefore, (RW) is also satisfied by both closures.

We now give counter-examples for the unsatisfied properties:

5. (Or) - consider the defeasible KB $\mathcal{K} = \{ A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq \neg C,\ A \sqsubseteq D,\ G \sqsubseteq D,\ D \sqsubseteq E,\ G \sqsubseteq H,\ H \sqsubseteq \neg E,\ G \sqsubseteq E \}$. The ranking for $\mathcal{K}$ is: $\mathcal{D}_0 = \{ B \sqsubseteq C,\ D \sqsubseteq E,\ H \sqsubseteq \neg E \}$, $\mathcal{D}_1 = \{ A \sqsubseteq B,\ A \sqsubseteq \neg C,\ A \sqsubseteq D,\ G \sqsubseteq D,\ G \sqsubseteq H \}$. We can derive both $A \sqsubseteq E$ and $G \sqsubseteq E$, using basic and minimal Relevant Closure, from $\mathcal{K}$. The former conclusion can be inferred by noting that there is a single A-justification: $\{ A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq \neg C \}$, and the latter conclusion can be inferred by noting that the G-justifications are: $\{ G \sqsubseteq E,\ G \sqsubseteq H,\ H \sqsubseteq \neg E \}$ and $\{ G \sqsubseteq D,\ G \sqsubseteq H,\ D \sqsubseteq E,\ H \sqsubseteq \neg E \}$. Finally, it is clear that we *cannot* derive $A \sqcup G \sqsubseteq E$ because the $A \sqcup G$-justifications are: $\{ A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq \neg C,\ G \sqsubseteq E,\ G \sqsubseteq H,\ H \sqsubseteq \neg E \}$ and $\{ A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq \neg C,\ G \sqsubseteq D,\ G \sqsubseteq H,\ D \sqsubseteq E,\ H \sqsubseteq \neg E \}$.

6. (CM) & (RM) - consider the defeasible KB $\mathcal{K} = \{ E \sqsubseteq \neg G,\ H \sqsubseteq E,\ B \sqsubseteq \neg D,\ C \sqsubseteq D,\ C \sqsubseteq B,\ C \sqcap D \sqsubseteq G,\ C \sqsubseteq H,\ C \sqcap D \sqsubseteq H \}$. The ranking for $\mathcal{K}$ is: $\mathcal{D}_0 = \{ E \sqsubseteq \neg G,\ B \sqsubseteq \neg D,\ H \sqsubseteq \neg E \}$, $\mathcal{D}_1 = \{ C \sqsubseteq D,\ C \sqsubseteq B,\ C \sqcap D \sqsubseteq G,\ C \sqsubseteq H,\ C \sqcap D \sqsubseteq H \}$. There is only one C-justification $\{ B \sqsubseteq \neg D,\ C \sqsubseteq D,\ C \sqsubseteq B \}$ and it is clear from this that one can derive both $C \sqsubseteq D$ and $C \sqsubseteq E$, using both basic and minimal Relevant Closure. However, we have three $(C \sqcap D)$-justifications $\{ B \sqsubseteq \neg D,\ C \sqsubseteq D,\ C \sqsubseteq B \}$, $\{ C \sqcap D \sqsubseteq G,\ C \sqcap D \sqsubseteq H,\ H \sqsubseteq E,\ E \sqsubseteq \neg G \}$ and $\{ C \sqcap D \sqsubseteq G,\ C \sqsubseteq H,\ H \sqsubseteq E,\ E \sqsubseteq \neg G \}$. Clearly, one cannot derive $C \sqcap D \sqsubseteq E$ after removing the $(C \sqcap D)$-basis axioms from $\mathcal{D}_0$. □

5.2 Non-Preferential Algorithms

In this section we evaluate whether the main alternative formalisms for defeasible reasoning in DLs satisfy the KLM properties or not. It is critical to point out that, by design, most of these alternatives cannot be *directly* compared to the preferential approach, nor *directly* evaluated against the KLM properties discussed in Section 3.2.1 of Chapter 3.

This is because in many cases the applicable formalism has been designed towards addressing different concrete reasoning problems than our preferential reasoning approaches. Therefore, we have to evaluate some of the formalisms by first placing some restrictions and underlying assumptions on the representations we choose, so that they can be directly evaluated against the KLM consequence relation postulates introduced in Chapter 3.

We address the formalisms in *order* of how *directly* they can be evaluated against the KLM postulates (i.e., how many restrictions do we need to place on the formalism before it can be directly evaluated). We start with the formalism which can be *most* directly evaluated - overriding (Section 2.9).

We do not consider DLs of MKNF and Defeasible Logic in this evaluation. The main reason for eliminating DLs of MKNF is that they differ quite considerably from the aforementioned formalisms in terms of their representational goals (they are more general formalisms allowing one to express epistemic statements in addition to default statements). We also eliminate Defeasible Logic because, when integrated into DLs, it is too dissimilar in terms of its approach, to the aforementioned formalisms.

5.2.1 Overriding

The most recent approach currently to defeasible reasoning for DLs, is the one by Bonatti et al. [27] on overriding (see Section 2.9). It is specifically targetted at DLs and is, representationally speaking, most similar to the preferential approach.

The central representational element introduced is the notion of a defeasible inclusion $\sqsubseteq_n$ relation (analogous in spirit to $\precsim$). Because of this correspondence, we can consider $\sqsubseteq_n$ to induce a consequence relation on DL concepts in an analogous way to $\precsim$.

We recall that a major difference between preferential reasoning, and other defeasible formalisms for DLs, is that specificity (see Example 12 on Page 62) is internalised in the mechanism of preferential reasoning (i.e., specificity is natively respected). Whereas, in other formalisms, priorities between

defaults can either be user-specified or be assumed to be defined by specificity [137, 12, 33].

The version of overriding we talk about here is the one which assumes that priorities among defeasible inclusions are defined by specificity. Bonatti et al. have made our work much easier by evaluating their formalism against the KLM rationality properties themselves [27, Theorem 8, p30].

However, there are two issues we have identified with their evaluation that need to be addressed in this thesis. The first is that the authors have not considered the (And) property in their presentation. The second is that the authors have considered, for some KLM properties, two possible formulations of these properties and evaluated them in both. Here will extend the insight of Bonatti et al. by demonstrating whether or not the (And) property holds for overriding. For the latter issue we will make clear which KLM properties have more than one formulation for overriding and which of these formulations are applicable for our evaluation. The (And) property can formulated as follows in the setting of overriding:

$$\text{(And)}\ \frac{C \sqsubseteq_n D,\ C \sqsubseteq_n E}{C \sqsubseteq_n D \sqcap E}$$

We now demonstrate that (And) is actually satisfied in general by overriding:

Lemma 25 (Overriding w.r.t. KLM Postulate (And)) *Overriding satisfies (And).*

Proof: We have to show that, given a $\mathcal{DL}^N$ KB $\mathcal{K}$ and concepts C, D and E, if $\mathcal{K} \mid\!\approx C \sqsubseteq_n D$ and $\mathcal{K} \mid\!\approx C \sqsubseteq_n E$ then $\mathcal{K} \mid\!\approx C \sqsubseteq_n D \sqcap E$. We assume that the premises hold but that $\mathcal{K} \not\mid\!\approx C \sqsubseteq_n D \sqcap E$. We try to derive a contradiction from this. We pick a model $\mathcal{I}$ for $\mathcal{K}$ s.t. $\mathcal{I} \not\Vdash C \sqsubseteq_n D \sqcap E$. From Definition 25 on Page 104 we know that there is a normality concept NX s.t. NX does *not* override $C \sqsubseteq_n D \sqcap E$ in $\mathcal{I}$ *and* that there is an $x \in \text{N}X^{\mathcal{I}}$ s.t. $x \in C^{\mathcal{I}}$ but that $x \notin (D \sqcap E)^{\mathcal{I}}$. Because N$X$ does *not* override $C \sqsubseteq_n D \sqcap E$ in $\mathcal{I}$ we know from Definition 24 on Page 101 that there is a

pre-model $\mathcal{J}$ for $\mathcal{K}$ s.t. $\mathrm{N}X^{\mathcal{J}} \neq \emptyset$, for each $x \in \mathrm{N}X^{\mathcal{J}}$, $x \in (\neg C \sqcup (D \sqcap E))^{\mathcal{J}}$, and for all higher priority DIs condition 3 of Definition 24 also holds. But we also know that $\mathcal{I} \Vdash C \sqsubseteq_n D$ and $\mathcal{I} \Vdash C \sqsubseteq_n E$ because $\mathcal{I}$ is a model for $\mathcal{K}$ and $\mathcal{K} \mathrel{|\!\approx} C \sqsubseteq_n D$ and $\mathcal{K} \mathrel{|\!\approx} C \sqsubseteq_n E$. By Definition 25 on Page 104 we know that for each normality concept $\mathrm{N}Y$ either $\mathrm{N}Y$ overrides $C \sqsubseteq_n D$ (resp. $C \sqsubseteq_n E$) in $\mathcal{I}$ or each $x \in \mathrm{N}Y^{\mathcal{I}}$ is s.t. $x \in (\neg C \sqcup D)^{\mathcal{I}}$ (resp. $(\neg C \sqcup E)^{\mathcal{I}}$). Suppose that $\mathrm{N}X$ overrides $C \sqsubseteq_n D$ (resp. $C \sqsubseteq_n E$) in $\mathcal{I}$. This means that there is *no* pre-model $\mathcal{J}'$ for $\mathcal{K}$ s.t. all the conditions of Definition 24 are satisfied for these DIs w.r.t. $\mathrm{N}X$ in $\mathcal{I}$. But it is clear that $(\neg C \sqcup (D \sqcap E))^{\mathcal{J}} \subseteq (\neg C \sqcup D)^{\mathcal{J}}$ and $(\neg C \sqcup (D \sqcap E))^{\mathcal{J}} \subseteq (\neg C \sqcup E)^{\mathcal{J}}$ (i.e., the first two conditions *are* satisfied for these DIs w.r.t. $\mathrm{N}X$ in $\mathcal{J}$). Therefore it must be the case that the pre-model $\mathcal{J}$ does not satisfy the *third* condition for overriding for these DIs w.r.t. $\mathrm{N}X$ in $\mathcal{I}$. This means that there is a DI δ' which is of higher priority than these DIs which is sacrificed even though it is not explicitly overridden in $\mathcal{I}$. But the DIs that are of higher priority than $C \sqsubseteq_n D$ and $C \sqsubseteq_n E$ are the *same* as those that are of higher priority than $C \sqsubseteq_n D \sqcap E$ (because priority is defined by specificity of the DI antecedents). But this would mean that condition 3 of Definition 24 does *not* hold for $C \sqsubseteq_n D \sqcap E$ w.r.t. $\mathrm{N}X$ in $\mathcal{I}$. This is a contradiction with our assumption that there is a pre-model $\mathcal{J}$ for $\mathcal{K}$ satisfying all the overriding conditions for $C \sqsubseteq_n D \sqcap E$. Therefore, there is no such pre-model and $C \sqsubseteq_n D \sqcap E$ is satisfied in $\mathcal{I}$, and hence $\mathcal{K} \mathrel{|\!\approx} C \sqsubseteq_n D \sqcap E$. □

Finally, we can clarify that, although Bonatti et al. claim that (LLE) is not satisfied in general by overriding [27, Theorem 8, p30], this result holds for the following formulation of the KLM property:

$$\text{(LLE1)}\ \frac{C \sqsubseteq_n E,\ C \equiv D}{D \sqsubseteq_n E}$$

That is, in the above formulation of (LLE), $C \equiv D$ is interpreted as defeasible consequence rather than as a classical consequence of the (strict part of the)

KB. Another possible interpretation of this property is to treat $C \equiv D$ as a classical consequence of the strict component of the KB (just the classical subsumptions). The property can then be formulated as follows.

$$\text{(LLE2)}\ \frac{C \sqsubseteq_n E, \models C \equiv D}{D \sqsubseteq_n E}$$

It turns out that (LLE2) is satisfied in general by overriding [27, Theorem 10, p35]. In an analogous way, one can also formulate two versions of the (RW) property for overriding. However, in that case, neither version holds in general for overriding and a counter-example is given to show this [27, Theorem 8, p30]. The following theorem consolidates the evaluation results for overriding (where priorities among DIs are defined by specificity).

Theorem 3 (Overriding w.r.t. KLM Postulates) *Overriding satisfies the properties (Ref), (Or), (And) and (LLE2) but does not satisfy (LLE1), (RW), (CM) and (RM).*

5.2.2 Circumscription

As there are a variety of semantics for circumscription in DLs, and there are no standardised circumscription patterns, we have to choose a representative semantics and pattern which can be evaluated against the KLM postulates.

Here we subscribe to the semantics of Bonatti, Lutz and Wolter [32] because it defines a basic, intuitive and representative semantics for circumscription in DLs. We have stated the semantics in Section 2.4.1 which expresses three conditions for determining if a model $\mathcal{I}$ for a circumscribed KB is more *preferred* or more *minimal* than another model $\mathcal{J}$ for the KB.

In that same section we talked about a fourth possible condition for when we would like to fix the extension of certain predicates. Here we do not consider this fourth condition. In fact, in terms of circumscription pattern, we are not going to consider varying or fixing predicates. We only introduce abnormality predicates that are to be minimised during reasoning.

We are going to evaluate this version of circumscription against the KLM postulates *regardless of varying or fixing other predicates*. This is helpful because we can get a more *general* perspective of the inferential character of circumscription independent of user-specified circumscription patterns.

From a representational perspective, in order to formulate the KLM postulates within this setting we have established, we define a translation of defeasible subsumptions of the form $C \sqsubseteq\!\!\!\!\sim D$ to *circumscription defaults* of the form $C \sqsubseteq D \sqcup Ab_C$.

Definition 55 (Circumscription Default) *A* circumscription default *is a subsumption of the form $C \sqsubseteq D \sqcup Ab_C$ where C and D are $\mathcal{ALC}$ concepts and Ab_C is a concept name representing the abnormal instances of C.*

Of course, we do not claim that circumscription defaults capture the same semantics as $C \sqsubseteq\!\!\!\!\sim D$ (indeed circumscription defaults are interpreted in classical DL interpretations while defeasible subsumptions are interpreted in ranked interpretations). Rather, we believe that the notion of circumscription default is a natural analogue of defeasible subsumption in the circumscription setting and intuitively captures a similar meaning in the latter context.

Therefore, even though circumscription does not introduce a defeasible subsumption relation in the language like $\sqsubseteq\!\!\!\!\sim$ (preferential reasoning) and $\sqsubseteq_n$ (overriding), we can actually *interpret* $\sqsubseteq$ in the circumscription default $C \sqsubseteq D \sqcup Ab_C$ as defining a notion of defeasible subsumption because we are going to minimise Ab_C during reasoning. That is, this syntactic form of subsumption statement represents a similar intuitive meaning to $C \sqsubseteq\!\!\!\!\sim D$ and hence allows us to reformulate the KLM postulates for this new notion of defeasible subsumption, and to evaluate consequence in circumscription against these postulates.

However, before we formulate the KLM postulates in this context, we have to define the kinds of KBs we are interested in for this version of circumscription, as well as the general circumscription pattern we consider. For the former, we are going to focus on *circumscribed defeasible KBs*:

Definition 56 (Circumscribed Defeasible KB) *A* circumscribed defeasible KB *is a structure of the form* $\langle \mathcal{T}, \mathcal{D} \rangle$ *where* $\mathcal{T}$ *is a classically consistent set of subsumptions of the form* $C \sqsubseteq D$ *s.t.* C *and* D *do not contain any abnormality predicates, and* $\mathcal{D}$ *is a set of circumscription defaults (Definition 55 on Page 244).*

Given a circumscribed defeasible KB, when performing reasoning, we only consider a very general circumscription pattern. As we discussed in Section 2.4.2, we define a partial order on the abnormality predicates ocurring in $\mathcal{D}$ determined by *specificity*. Specificity in this context means that an abnormality predicate Ab_{C1} is given minimisation priority during reasoning over another predicate Ab_{C2} if: $\mathcal{T} \cup \mathcal{D} \models C_1 \sqsubseteq C_2$ and $\mathcal{T} \cup \mathcal{D} \not\models C_2 \sqsubseteq C_1$. That is, we admit only the following circumscription pattern (called a *basic circumscription pattern*):

Definition 57 (Basic Circumscription Pattern) *Let* $\langle \mathcal{T}, \mathcal{D} \rangle$ *be a circumscribed defeasible KB and* δ *a circumscription default. Then, the* basic circumscription pattern *w.r.t.* $\langle \mathcal{T}, \mathcal{D} \rangle$ *and* δ *is the structure* $\langle \mathcal{M}, \prec_{\mathcal{M}} \rangle$ *where* $\mathcal{M}$ *is the set of all abnormality predicates* Ab_C *appearing in* $\mathcal{D}$ *together with the one in* δ*, and* $\prec_{\mathcal{M}}$ *is a strict partial order on the elements of* $\mathcal{M}$ *defined by specificity. The elements of* $\mathcal{M}$ *are to be minimised during reasoning and* $\prec_{\mathcal{M}}$ *defines the minimisation priority among the predicates of* $\mathcal{M}$*.*

Notice that we do not allow the minimisation of roles in this form of circumscription. Bonatti et al. [32] refer to this kind of circumscription as *concept circumscription*. Entailment in circumscription is defined in terms of *minimal models*. That is, given a circumscribed defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle$, there is an ordering $<_{min}$ placed on the standard DL models of $\mathcal{T} \cup \mathcal{D}$. The ordering is defined by the three conditions mentioned earlier in this section and in Section 2.4.1. A model $\mathcal{I}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ is thus a minimal model for $\langle \mathcal{T}, \mathcal{D} \rangle$ if there is no other model $\mathcal{J}$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. $\mathcal{J} <_{min} \mathcal{I}$. The notation of entailment is presented in the following definition:

Definition 58 (Entailment in Circumscription) *Let $\langle \mathcal{T}, \mathcal{D} \rangle$ be a circumscribed defeasible KB and δ a circumscription default. Then, δ is entailed by $\langle \mathcal{T}, \mathcal{D} \rangle$, written as $\langle \mathcal{T}, \mathcal{D} \rangle \models_{circ} \delta$, if each minimal model for $\langle \mathcal{T}, \mathcal{D} \rangle$ is a model for δ.*

We can now reformulate the KLM postulates within this setting:

Definition 59 (KLM Rationality Postulates for Circumscription) *The KLM rationality postulates for circumscription are as follows:*

$$(Ref) \quad \models C \sqsubseteq C \sqcup Ab_C$$

$$(LLE) \quad \frac{\models C \equiv D, \models C \sqsubseteq E \sqcup Ab_C}{\models D \sqsubseteq E \sqcup Ab_D}$$

$$(And) \quad \frac{\models C \sqsubseteq D \sqcup Ab_C, \models C \sqsubseteq E \sqcup Ab_C}{\models C \sqsubseteq (D \sqcap E) \sqcup Ab_C}$$

$$(Or) \quad \frac{\models C \sqsubseteq E \sqcup Ab_C, \models D \sqsubseteq E \sqcup Ab_D}{\models C \sqcup D \sqsubseteq E \sqcup Ab_{C \sqcup D}}$$

$$(RW) \quad \frac{\models C \sqsubseteq D \sqcup Ab_C, \models D \sqsubseteq E}{\models C \sqsubseteq E \sqcup Ab_C}$$

$$(CM) \quad \frac{\models C \sqsubseteq D \sqcup Ab_C, \models C \sqsubseteq E \sqcup Ab_C}{\models C \sqcap D \sqsubseteq E \sqcup Ab_{C \sqcap D}}$$

$$(RM) \quad \frac{\models C \sqsubseteq E \sqcup Ab_C, \not\models C \sqsubseteq \neg D \sqcup Ab_C}{\models C \sqcap D \sqsubseteq E \sqcup Ab_{C \sqcap D}}$$

It proves to be quite straightforward to evaluate the form of circumscription we have defined, which we shall call *basic circumscription*, against the properties formulated in Definition 59 on Page 246. The result is demonstrated by the proof of the following theorem:

Theorem 4 (Basic Circumscription w.r.t. KLM Postulates) *Basic circumscription satisfies the properties (Ref), (LLE), (RW), (And) and (Or) but does not satisfy (CM) and (RM).*

Proof: we show that for any circumscribed defeasible KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{D} \rangle$ and concepts C, D and E:

1. (Ref) - $\mathcal{K} \models_{circ} C \sqsubseteq C \sqcup Ab_C$. This is clearly straightforward because $C^{\mathcal{I}} \subseteq C^{\mathcal{I}} \cup {Ab_C}^{\mathcal{I}}$ for any interpretation $\mathcal{I}$ for any KB and C.

2. (And) - if $\mathcal{K} \models_{circ} C \sqsubseteq D \sqcup Ab_C$ and $\mathcal{K} \models_{circ} C \sqsubseteq E \sqcup Ab_C$ then $\mathcal{K} \models_{circ} C \sqsubseteq (D \sqcap E) \sqcup Ab_C$. We pick a $\mathcal{K}$, C, D and E s.t. the premises are satisfied. Picking a minimal model $\mathcal{I}$ for $\mathcal{K}$, since $\mathcal{I} \Vdash C \sqsubseteq D \sqcup Ab_C$ and $\mathcal{I} \Vdash C \sqsubseteq E \sqcup Ab_C$, we know that $\mathcal{I} \Vdash C \sqsubseteq (D \sqcup Ab_C) \sqcap (E \sqcup Ab_C)$. By the distributive law of set algebra, we can therefore conclude that $\mathcal{I} \Vdash C \sqsubseteq (D \sqcap E) \sqcup Ab_C$ and therefore that $\mathcal{K} \models_{circ} C \sqsubseteq (D \sqcap E) \sqcup Ab_C$.

3. (LLE) - if $\mathcal{T} \cup \mathcal{D} \models C \equiv D$ and $\mathcal{K} \models_{circ} C \sqsubseteq E \sqcup Ab_C$ then $\mathcal{K} \models_{circ} D \sqsubseteq E \sqcup Ab_D$. We pick a $\mathcal{K}$, C, D and E s.t. the premises are satisfied. We also pick a minimal model $\mathcal{I}$ for $\mathcal{K}$. We know that $\mathcal{I} \Vdash C \sqsubseteq E \sqcup Ab_C$ and $\mathcal{I} \Vdash C \equiv D$. I.e., that $C^{\mathcal{I}} \subseteq E^{\mathcal{I}} \cup {Ab_C}^{\mathcal{I}}$ and $C^{\mathcal{I}} = D^{\mathcal{I}}$. We have to show that $D^{\mathcal{I}} \subseteq E^{\mathcal{I}} \cup {Ab_D}^{\mathcal{I}}$. In other words, it would be sufficient to show that ${Ab_C}^{\mathcal{I}} \subseteq {Ab_D}^{\mathcal{I}}$. Assume that ${Ab_C}^{\mathcal{I}} \not\subseteq {Ab_D}^{\mathcal{I}}$. This means there is an $x \in {Ab_C}^{\mathcal{I}}$ s.t. $x \notin {Ab_D}^{\mathcal{I}}$.

Case 1: $x \in C^{\mathcal{I}}$. Therefore, $x \in D^{\mathcal{I}}$ because $C^{\mathcal{I}} = D^{\mathcal{I}}$. Also, $x \in E^{\mathcal{I}} \cup {Ab_D}^{\mathcal{I}}$ because $D^{\mathcal{I}} \subseteq E^{\mathcal{I}} \cup {Ab_D}^{\mathcal{I}}$. From our assumption that $x \notin {Ab_D}^{\mathcal{I}}$ it must be the case that $x \in E^{\mathcal{I}}$. But this means that $\mathcal{I}$ is not a minimal model for $\mathcal{K}$ because even if $x \notin {Ab_C}^{\mathcal{I}}$, $\mathcal{I}$ will still be a model for $\mathcal{K}$. This is a contradiction with our assumptions.

Case 2: $x \notin C^{\mathcal{I}}$. This also means that $\mathcal{I}$ is not a minimal model for $\mathcal{K}$ because even if $x \notin {Ab_C}^{\mathcal{I}}$, $\mathcal{I}$ will still be a model for $\mathcal{K}$ (because $C^{\mathcal{I}} \subseteq E^{\mathcal{I}} \cup {Ab_C}^{\mathcal{I}}$). This is a contradiction with our assumptions. Therefore, ${Ab_C}^{\mathcal{I}} \subseteq {Ab_D}^{\mathcal{I}}$ and hence $D^{\mathcal{I}} \subseteq E^{\mathcal{I}} \cup {Ab_D}^{\mathcal{I}}$. Finally, $\mathcal{K} \models_{circ} D \sqsubseteq E \sqcup Ab_D$.

4. (RW) - if $\mathcal{T} \cup \mathcal{D} \models D \equiv E$ and $\mathcal{K} \models_{circ} C \sqsubseteq D \sqcup Ab_C$ then $\mathcal{K} \models_{circ} C \sqsubseteq E \sqcup Ab_C$. We pick a $\mathcal{K}$, C, D and E s.t. the premises are satisfied. We also pick a minimal model $\mathcal{I}$ for $\mathcal{K}$. It is clear that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}} \cup {Ab_C}^{\mathcal{I}}$ and $D^{\mathcal{I}} \subseteq E^{\mathcal{I}}$. Hence, by union of sets, it is clear that $C^{\mathcal{I}} \subseteq E^{\mathcal{I}} \cup {Ab_C}^{\mathcal{I}}$. Finally, we can conclude that $\mathcal{K} \models_{circ} C \sqsubseteq E \sqcup Ab_C$.

5. (Or) - it must be said that (Or) is not satisfied in general for arbitrary priority relations among abnormality predicates. To see this, consider the following circumscribed defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle = \langle \{\top \sqsubseteq (C \sqcup E) \sqcap \neg D\}, \{C \sqsubseteq D \sqcup Ab_C, E \sqsubseteq D \sqcup Ab_E\} \rangle$ and the circumscription default $C \sqcup E \sqsubseteq D \sqcup Ab_{C \sqcup E}$. Let $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ be an interpretation s.t. $\Delta^{\mathcal{I}} = \{x\}$ and $\cdot^{\mathcal{I}}$ is defined s.t. ${C^{\mathcal{I}}} = {Ab_C}^{\mathcal{I}} = \{x\}$ and ${E^{\mathcal{I}}} = {D^{\mathcal{I}}} = {Ab_E}^{\mathcal{I}} = {Ab_{C \sqcup E}}^{\mathcal{I}} = \emptyset$. It is clear that $\mathcal{I}$ is a model for $\mathcal{T} \cup \mathcal{D}$ and, if we fix the non-abnormality predicates, it is also clear that there is no $\mathcal{J} <_{min} \mathcal{I}$ s.t. $\mathcal{J}$ is a model for $\mathcal{T} \cup \mathcal{D}$. In other words, it is clear that the extension of Ab_C could not be empty in $\mathcal{J}$ because this would violate the default $C \sqsubseteq D \sqcup Ab_C$. Therefore, $\mathcal{I}$ is a minimal model for $\langle \mathcal{T}, \mathcal{D} \rangle$, $\mathcal{I} \Vdash C \sqsubseteq D \sqcup Ab_C$, $\mathcal{I} \Vdash E \sqsubseteq D \sqcup Ab_E$ but $\mathcal{I} \nVdash C \sqcup E \sqsubseteq D \sqcup Ab_{C \sqcup E}$ because $x \in (C \sqcup E)^{\mathcal{I}}$ and $x \notin D^{\mathcal{I}}$ and $x \notin {Ab_{C \sqcup E}}^{\mathcal{I}}$. However, in the given counter-example, specificity is not respected during minimisation of abnormality predicates. Since we are interested only in priority relations defined by specificity we can demonstrate that (Or) is satisfied with this restriction. We have to show that if $\mathcal{K} \models_{circ} C \sqsubseteq D \sqcup Ab_C$ and $\mathcal{K} \models_{circ} E \sqsubseteq D \sqcup Ab_E$ then $\mathcal{K} \models_{circ} C \sqcup E \sqsubseteq D \sqcup Ab_{C \sqcup E}$. We pick a $\mathcal{K}$, C, D and E s.t. the premises are satisfied. We also pick a minimal model $\mathcal{I}$ for $\mathcal{K}$. It is clear that $\mathcal{I} \Vdash C \sqsubseteq D \sqcup Ab_C$ and $\mathcal{I} \Vdash E \sqsubseteq D \sqcup Ab_E$. But we also know, because of specificity, that ${Ab_C}^{\mathcal{I}} \subseteq {Ab_{C \sqcup E}}^{\mathcal{I}}$ and ${Ab_E}^{\mathcal{I}} \subseteq {Ab_{C \sqcup E}}^{\mathcal{I}}$. Let us assume that $\mathcal{I} \nVdash C \sqcup E \sqsubseteq D \sqcup Ab_{C \sqcup E}$. We will derive a contradiction from this. It means that there is an $x \in (C \sqcup E)^{\mathcal{I}}$ s.t. $x \notin D^{\mathcal{I}}$ and $x \notin {Ab_{C \sqcup E}}^{\mathcal{I}}$.
Case 1: $x \in C^{\mathcal{I}}$. This means that $x \in {Ab_C}^{\mathcal{I}}$ because $\mathcal{I} \Vdash C \sqsubseteq D \sqcup Ab_C$ and $x \notin D^{\mathcal{I}}$. But specificity tells us that ${Ab_C}^{\mathcal{I}} \subseteq {Ab_{C \sqcup E}}^{\mathcal{I}}$ (because $C \sqsubseteq C \sqcup E$ is a tautology) and therefore that $x \in {Ab_{C \sqcup E}}^{\mathcal{I}}$. This is a contradiction. Therefore it must be the case that $x \in E^{\mathcal{I}}$
Case 2: $x \in E^{\mathcal{I}}$. We follow an analogous argument to Case 1 to find that $x \in {Ab_{C \sqcup E}}^{\mathcal{I}}$ and derive a contradiction again.

We therefore have to conclude that $\mathcal{I} \Vdash C \sqcup E \sqsubseteq D \sqcup Ab_{C \sqcup E}$ and hence that $\mathcal{K} \models_{circ} C \sqcup E \sqsubseteq D \sqcup Ab_{C \sqcup E}$.

We now give counter-examples to demonstrate that basic circumscription does *not* satisfy (CM) and (RM). The strategy is that, for each property, we identify a circumscribed defeasible KB s.t. the premises of the property are entailed by this KB and the conclusion is *not* (using Definition 58 on Page 246).

6. (CM) - Consider the circumscribed defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle = \langle \{\top \sqsubseteq C \sqcap E \sqcap \neg D\}, \{C \sqsubseteq D \sqcup Ab_C, C \sqsubseteq E \sqcup Ab_C\} \rangle$ and the circumscription default $C \sqcap E \sqsubseteq D \sqcup Ab_{C \sqcap E}$. Let $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ be an interpretation s.t. $\Delta^{\mathcal{I}} = \{x\}$ and $\cdot^{\mathcal{I}}$ is defined s.t. $C^{\mathcal{I}} = {Ab_C}^{\mathcal{I}} = E^{\mathcal{I}} = \{x\}$ and $D^{\mathcal{I}} = {Ab_{C \sqcap E}}^{\mathcal{I}} = \emptyset$. It is clear that $\mathcal{I}$ is a model for $\mathcal{T} \cup \mathcal{D}$ and it is also clear that there is no $\mathcal{J} <_{min} \mathcal{I}$ s.t. $\mathcal{J}$ is a model for $\mathcal{T} \cup \mathcal{D}$. In other words, it is clear that the extension of Ab_C could not be empty in $\mathcal{J}$ because this would violate the default $C \sqsubseteq D \sqcup Ab_C$. Therefore, $\mathcal{I}$ is a minimal model for $\langle \mathcal{T}, \mathcal{D} \rangle$ and $\mathcal{I} \not\Vdash C \sqcap E \sqsubseteq D \sqcup Ab_{C \sqcap E}$ because $x \in (C \sqcap E)^{\mathcal{I}}$ and $x \notin D^{\mathcal{I}}$ and $x \notin {Ab_{C \sqcap E}}^{\mathcal{I}}$.
7. (RM) - Consider the circumscribed defeasible KB $\langle \mathcal{T}, \mathcal{D} \rangle = \langle \emptyset, \{C \sqsubseteq D \sqcup Ab_C\} \rangle$ and the circumscription default $C \sqcap E \sqsubseteq D \sqcup Ab_{C \sqcap E}$. Suppose we minimise Ab_C and $Ab_{C \sqcap E}$ and *fix* the rest of the predicates. It is clear that $\mathcal{K} \models_{circ} C \sqsubseteq D \sqcup Ab_C$ and that $\mathcal{K} \not\models_{circ} C \sqsubseteq \neg E \sqcup Ab_C$. We show that $\mathcal{K} \not\models_{circ} C \sqcap E \sqsubseteq D \sqcup Ab_{C \sqcap E}$. Consider the following interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ where $\Delta^{\mathcal{I}} = \{x, y\}$ and $\cdot^{\mathcal{I}}$ is defined s.t. $C^{\mathcal{I}} = E^{\mathcal{I}} = \{x, y\}$, $D^{\mathcal{I}} = \{x\}$, ${Ab_C}^{\mathcal{I}} = \{y\}$, $Ab^{\mathcal{I}}_{C \sqcap E} = \emptyset$. Clearly, $\mathcal{I}$ is a model for $\mathcal{K}$. Assume that $\mathcal{I}$ is not a *minimal* model for $\mathcal{K}$. This can only mean that there is a model $\mathcal{J}$ for $\mathcal{K}$ which is the same as $\mathcal{I}$ except for the extension of Ab_C. That is, s.t. ${Ab_C}^{\mathcal{J}} \subset {Ab_C}^{\mathcal{I}}$, or more specifically, s.t. ${Ab_C}^{\mathcal{J}} = \emptyset$. But this cannot be the case because such a $\mathcal{J}$ would no longer satisfy $C \sqsubseteq D \sqcup Ab_C$, because $y \in C^{\mathcal{J}}$ and $y \notin D^{\mathcal{J}}$ and $y \notin {Ab_C}^{\mathcal{J}}$.Therefore, $\mathcal{I}$ is a minimal model for $\mathcal{K}$ and it is clear that $\mathcal{I} \not\Vdash C \sqcap E \sqsubseteq D \sqcup Ab_{C \sqcap E}$ because $y \in (C \sqcap E)^{\mathcal{I}}$ and $y \notin D^{\mathcal{I}}$ and $y \notin {Ab_{C \sqcap E}}^{\mathcal{I}}$. Hence, $\mathcal{K} \not\models_{circ} C \sqcap E \sqsubseteq D \sqcup Ab_{C \sqcap E}$. □

Even though we do not explicitly take priorities among abnormality predicates into consideration in our proofs, the properties that hold in general would also hold even if prioritisation is defined by specificity. This is because we have given a very general definition for minimising abnormality predicates (i.e., for *minimal models* of a circumscribed KB). Therefore, by taking a view independent of circumscription pattern, we are allowed to substitute any such pattern to define a notion of minimal model. This means that we consider *all* possible minimal models in the broadest sense of the word *all*. When we incorporate priorities, some minimal models will obviously no longer be minimal and hence we are considering a subset of these. That is, if our results hold for *all* minimal models it will hold for a subset of these.

For properties (CM) and (RM) that don't hold, our counter-examples are also applicable when considering priorities defined by specificity. This is because the given counter-models are still minimal models when considering the priorities among abnormality predicates. Thus the results we have obtained still apply to cases of prioritisation among abnormality predicates.

5.2.3 Default Reasoning

For classical default logic, the broad focus has always been on applying a set of defaults to a set of first-order logic formulas to extend this set with new first-order formulas representing defeasible inferences that one can make from the default theory. As we demonstrated in Section 2.5, current adaptations of Reiter's default logic to the DL case define an analogous mechanism [11]. The representational differences are that our default theory in the DL case consists of normal defaults of the form $\frac{C:D}{D}$ where C and D are DL concepts, and our set of first-order formulas is replaced by a set of DL axioms (either TBox or ABox).

However, the issue with these approaches concerns the reasoning question being asked. That is, these approaches only consider extending the DL KB with *ABox* information. I.e., they define a way to apply the defaults to the given DL KB in order to defeasibly derive *ABox* statements from this

KB. Since our focus in this thesis is purely on *TBox* defeasible reasoning, an evaluation of the current default logic approaches for DLs against the KLM postulates is irrelevant.

Therefore, to make such an evaluation relevant, we have to generalise the mechanism of default logic so that we can apply normal defaults to a DL *TBox* to extend this TBox with more *classical subsumption statements* representing the defeasible inferences one can make from this TBox. We believe this scenario to be the most natural adaptation of default logic to derive TBox knowledge from application of defaults. Nevertheless, we have to be careful not to define an approach that is too far removed from the natural mechanism of default logic because our goal is to evaluate the general default reasoning mechanism and not invent a novel variant of default logic.

We will now adapt the definitions for the main constructs in default logic to be able to derive TBox information from our default theories. Thereafter, we consider how to evaluate the reasoning mechanism against the KLM postulates. We adopt the same definition of *normal* DL default that Baader et al. [11] subscribe to. That is, a normal DL default is one of the form $\frac{C:D}{D}$ where C and D are DL concepts and the statement intuitively means that if an object is known to have property C, and it is consistent to believe that it has property D also, then it has property D. This leads us to define what a *DL default KB* is in our setting:

Definition 60 (DL Default KB) *A* DL default KB *is a structure* $\langle \mathcal{T}, \mathcal{D} \rangle$ *where* $\mathcal{T}$ *is a consistent and finite set of classical DL subsumption statements and* $\mathcal{D}$ *is a finite set of normal DL defaults.*

Now the core principle in the reasoning mechanism for default logic is the notion of *extension* of a default KB. Here we reinterpret this notion in the context of DL Default KBs and call it a *DL default extension*:

Definition 61 (DL Default Extension) *Let* $\langle \mathcal{T}, \mathcal{D} \rangle$ *be a DL default KB. Then,* $\mathcal{T}' \supseteq \mathcal{T}$ *is a* DL extension *of* $\langle \mathcal{T}, \mathcal{D} \rangle$ *if:* $\mathcal{T}'$ *is deductively closed and for each* $\frac{C:D}{D} \in \mathcal{D}$*, if* $\top \sqsubseteq C \sqcap \neg D \notin \mathcal{T}'$ *then* $C \sqsubseteq D \in \mathcal{T}'$*.*

Reiter also gives an operational definition for extensions in terms of *fixed points*. Baader et al. [11], in their extension of defaults to DLs, also adopt this definition. We give the natural interpretation of this constructive definition for our context here:

Definition 62 (DL Default Extension as Fixed Points) *Let $\mathcal{E}$ be a set of classical DL subsumption statements, and $\langle \mathcal{T}, \mathcal{D} \rangle$ a DL default KB. We define $\mathcal{E}_0 = \mathcal{T}$ and for all $i \geq 0$: $\mathcal{E}_{i+1} := \mathcal{E}_i \cup \{C \sqsubseteq D \mid \frac{C:D}{D} \in \mathcal{D}$ and $\top \sqsubseteq C \sqcap \neg D \notin \mathcal{E}\}$. Then, $\mathcal{E}$ is a* DL default extension *of $\langle \mathcal{T}, \mathcal{D} \rangle$ if: $\mathcal{E} = \bigcup_{i \geq 0} Th(\mathcal{E}_i)$ where $Th(S)$ stands for the deductive closure of the set of classical DL subsumptions S.*

To define entailment in default logic one has to consider the *skeptical* approach and at least two versions of a *credulous* one. The skeptical approach defines that a default $\frac{C:D}{D}$ defeasibly follows from (is logically entailed by) a DL default KB $\langle \mathcal{T}, \mathcal{D} \rangle$ if $C \sqsubseteq D$ appears in *each* DL default extension of $\langle \mathcal{T}, \mathcal{D} \rangle$. We shall write this version of entailment as $\langle \mathcal{T}, \mathcal{D} \rangle \vdash_s \frac{C:D}{D}$. The most obvious credulous approach is to logically entail $\frac{C:D}{D}$ if it appears in *at least one* extension of $\langle \mathcal{T}, \mathcal{D} \rangle$. We write this version of entailment as $\langle \mathcal{T}, \mathcal{D} \rangle \vdash_c \frac{C:D}{D}$.

Yet another credulous approach is to pick an arbitrary extension and take this extension to define entailment. We do not consider this latter form of entailment in our evaluation because it requires us to consider the *choice* of this extension and this is not appropriate if we want to get a more *general* perspective of the inferential behaviour of default logic. Therefore, from here on we assume the term *credulous* to refer to the former (more popular) notion.
Skeptical Entailment: Interestingly, for the version of default logic we have presented, we find that skeptical entailment (what follows in *each* extension) is monotonic. This is because a DL default extension of any DL default KB $\langle \mathcal{T}, \mathcal{D} \rangle$ is the deductive closure (over $\models$) of a superset of $\mathcal{T}$. Clearly, because $\models$ is monotonic, skeptical entailment in our version of default logic is also monotonic. This is perhaps not surprising if we recall the goal of the underlying mechanism of default logic and the notion of extension itself.

That is, essentially we are constructing *all* supersets of a classical theory that are internally consistent (by using a set of default rules as generative agents to catalyse this process).

Implied by this is the fact that, if we take the union of any two extensions of a default theory, we will get an inconsistency [161, Theorem 3.3]. Indeed, they are separate extensions exactly *because* of this fact. Because of this general situation, it is very straightforward to verify that this version of default logic satisfies *all* the KLM postulates and so we omit proofs for this case. But as our above discussion points out, this result is not necessarily a positive aspect of skeptical entailment in default logic seeing as it is monotonic.

Credulous Entailment: The credulous approach only requires one extension to contain the inference under consideration. This view of entailment is not necessarily monotonic and we can formulate and evaluate the KLM properties w.r.t. this entailment relation. The results of this evaluation are demonstrated by Theorem 5 on Page 253:

Theorem 5 (Default Logic w.r.t. KLM postulates)
Skeptical Default Logic is monotonic and satisfies all the KLM postulates, and Credulous Default Logic is monotonic and satisfies all the KLM postulates except for (And) and (Or).

Proof: The skeptical case is trivial, we prove the credulous case. We show for any DL default KB $\langle \mathcal{T}, \mathcal{D} \rangle$ and concepts C, D and E:
1. (Ref) - $\langle \mathcal{T}, \mathcal{D} \rangle \vdash_c \frac{C:C}{C}$. In other words, we have to show that there is a DL default extension $\mathcal{T}'$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. $C \sqsubseteq C \in \mathcal{T}'$. Since $\mathcal{T}'$ is deductively closed and $C \sqsubseteq C$ is a tautology it follows that (Ref) is satisfied.
2. (LLE) - if $\langle \mathcal{T}, \mathcal{D} \rangle \vdash_c \frac{C:D}{D}$ and $\mathcal{T} \models C \equiv E$ then $\langle \mathcal{T}, \mathcal{D} \rangle \vdash_c \frac{E:D}{D}$. Picking a $\langle \mathcal{T}, \mathcal{D} \rangle$ s.t. $\langle \mathcal{T}, \mathcal{D} \rangle \vdash_c \frac{C:D}{D}$ we can see that there is a DL default extension s.t. $C \sqsubseteq D$ is contained in this extension. The second premise $\mathcal{T} \models C \equiv E$ clearly demonstrates that $C \equiv E$ is contained in *all* extensions because every extension $\mathcal{T}'$ for $\langle \mathcal{T}, \mathcal{D} \rangle$ is s.t. $\mathcal{T}' \supseteq \mathcal{T}$. Therefore it is clear that $E \sqsubseteq D$ will be in the extension that $C \sqsubseteq D$ (and $C \equiv E$) appears in.

3. (RW) - an analogous argument to the one for (LLE) can be used to prove that (RW) alo holds.
4. (M) - We will now show that credulous entailment in our default logic is actually monotonic, and therefore satisfies (CM) and (RM) as well, because these are weakened versions of classical monotonicity. Notice that for any $C \sqsubseteq D \in \mathcal{T}'$ where $\mathcal{T}'$ is a DL default extension, it must be the case that $C \sqcap E \sqsubseteq D \in \mathcal{T}'$ for any E because $\mathcal{T}'$ is a deductively closed set of classical DL subsumptions and classical DLs satisfy monotonicity.

We now give a counter-example for the (And) property:

5. (And) - Consider the DL default KB $\langle \mathcal{T}, \mathcal{D} \rangle = \langle \{E \sqsubseteq \neg D\}, \{\frac{C:D}{D}, \frac{C:E}{E}\} \rangle$. There are two extensions for $\langle \mathcal{T}, \mathcal{D} \rangle$, namely, $\mathcal{E}_1 = Th(\{E \sqsubseteq \neg D\}) \cup \{C \sqsubseteq D\}$ and $\mathcal{E}_2 = Th(\{E \sqsubseteq \neg D\}) \cup \{C \sqsubseteq E\}$ where $Th(S)$ denotes the deductive closure of the set of sentences S. Of course, it is clear that $C \sqsubseteq D \sqcap E$ is not an element of either extension.
6. (Or) - Consider the DL default KB $\langle \mathcal{T}, \mathcal{D} \rangle = \langle \{\top \sqsubseteq (E \sqcap \neg D) \sqcup (C \sqcap \neg D)\}, \{\frac{C:D}{D}, \frac{E:D}{D}\} \rangle$. There are two extensions for $\langle \mathcal{T}, \mathcal{D} \rangle$, namely, $\mathcal{E}_1 = Th(\{\top \sqsubseteq (E \sqcap \neg D) \sqcup (C \sqcap \neg D)\}) \cup \{C \sqsubseteq D\}$ and $\mathcal{E}_2 = Th(\{\top \sqsubseteq (E \sqcap \neg D) \sqcup (C \sqcap \neg D)\}) \cup \{E \sqsubseteq D\}$ where $Th(S)$ denotes the deductive closure of the set of sentences S. Of course, it is clear that $C \sqsubseteq D$ and $E \sqsubseteq D$ do not *both* occur in any particular extension. □

5.3 Discussion

In this chapter we showed that, apart from Lexicographic Closure, Rational Closure and Lexicographically-Relevant Closure, all other preferential and non-preferential formalisms studied in this thesis do not satisfy all the KLM postulates. We found that the Relevant Closures, overriding and basic circumscription do not satisfy (CM) and (RM). In addition, (Or) is not satis-

fied by the Relevant Closures although overriding and basic circumscription both satisfy (Or) when the priority relations among DIs (resp. abnormality predicates) are defined by specificity. Overriding, however, does not satisfy standard (LLE) and (RW) as well. In our opinion, such properties are more important to be satisfied by defeasible entailment since they contain tautologies as part of their premises.

We presented a natural interpretation of the mechanism of default logic for generating TBox statements (subsumptions) by applying DL defaults to a classical DL TBox. This version of default logic was found to define a monotonic entailment relation for both skeptical and credulous entailment. In the former entailment regime all the KLM postulates are satisfied, while in the latter (And) and (Or) are *not* satisfied in general.

It is important to note that the monotonicity of credulous entailment is quite different from the monotonicity of skeptical entailment in default logic. We notice that the former is monotonic on the meta-level (entailment level) but it is nonmonotonic on the object level. For example, given the DL default KB $\langle \mathcal{T}, \mathcal{D} \rangle = \langle \{\mathsf{Penguin} \sqsubseteq \mathsf{Bird}\}, \{\frac{\mathsf{Bird}:\exists \mathsf{hasAbility.Flying}}{\exists \mathsf{hasAbility.Flying}}\} \rangle$ we can derive an extension $\mathcal{T}'$ of $\langle \mathcal{T}, \mathcal{D} \rangle$ containing $\mathsf{Penguin} \sqsubseteq \exists \mathsf{hasAbility.Flying}$. Even after adding $\frac{\mathsf{Penguin}:\neg(\exists \mathsf{hasAbility.Flying})}{\neg(\exists \mathsf{hasAbility.Flying})}$ to $\mathcal{D}$, $\mathcal{T}'$ still remains an extension because we cannot apply $\frac{\mathsf{Penguin}:\neg(\exists \mathsf{hasAbility.Flying})}{\neg(\exists \mathsf{hasAbility.Flying})}$ to extend $\mathcal{T}'$ further. However, if we apply $\frac{\mathsf{Penguin}:\neg(\exists \mathsf{hasAbility.Flying})}{\neg(\exists \mathsf{hasAbility.Flying})}$ to penguins and not $\frac{\mathsf{Bird}:\exists \mathsf{hasAbility.Flying}}{\exists \mathsf{hasAbility.Flying}}$ we will derive that penguins do not fly, which overrides the flying ability of birds. This is clearly nonmonotonic behaviour on the object level. For skeptical entailment such behaviour is not allowed.

Of course, an interesting (though undesirable) behaviour of the credulous approach to default logic is that conflicting beliefs are allowed to coexist. For example, in the above KB $\langle \mathcal{T}, \mathcal{D} \rangle$, we can derive *both* $\langle \mathcal{T}, \mathcal{D} \rangle \vdash_c \frac{\mathsf{Penguin}:\neg(\exists \mathsf{hasAbility.Flying})}{\neg(\exists \mathsf{hasAbility.Flying})}$ and $\langle \mathcal{T}, \mathcal{D} \rangle \vdash_c \frac{\mathsf{Penguin}:\exists \mathsf{hasAbility.Flying}}{\exists \mathsf{hasAbility.Flying}}$ because there is an extension containing $\mathsf{Penguin} \sqsubseteq \exists \mathsf{hasAbility.Flying}$, one containing $\mathsf{Penguin} \sqsubseteq \neg(\exists \mathsf{hasAbility.Flying})$ and none containing $\mathsf{Penguin} \sqsubseteq \bot$.

Finally, it is worth mentioning that Reiter advocates the intended ap-

plication of default logic as determining *one* consistent set of beliefs about the world [161, Section 2.2]. That is, the intention is to pick a *single* extension and to reason within this extension until evidence forces one to switch to another. This scenario differs from the skeptical vs. credulous notions of reasoning we investigated in the previous section. It is clear that reasoning within this single extension is monotonic and satisfies all the KLM postulates but it is not clear as to how to choose such an extension (i.e., it appears to be a user-specified decision with no clear guidelines as how to choose it).

Chapter 6

Evaluating the Performance of Defeasible Reasoning

An important question that we ask of our defeasible reasoning algorithms, from a practical perspective, is: how much does one pay for the additional expressivity of defeasible subsumption in terms of practical reasoning performance. We have shown that the worst case computational complexity of Rational Closure is not higher than reasoning with the underlying classical formalism that we extend (see Section 4.3). This is good news, but does not guarantee good performance in practice.

On the other hand, we have also shown that Lexicographic Closure and its equivalent construction Lexicographically Relevant Closure are in the 2-EXPTIME complexity class. The same result holds for the Basic and Minimal Relevant Closures. In these cases, it would be interesting to observe whether this high complexity translates into exceptionally slow inferences in practice.

In this chapter we present experiments which attempt to answer these questions. The idea behind these experiments is to give a preliminary understanding of the practical performance one can expect from defeasible reasoning algorithms, when employed on "real world" ontologies. Also, as stated earlier, we wish to get a sense of how much more expensive (computationally speaking) defeasible reasoning is than classical reasoning for DLs.

As mentioned in Chapter 1, we have a significant problem when evaluating the performance of defeasible reasoning because there are no naturally occurring ontologies with defeasible features. That is, because the representation of defeasible knowledge is not yet natively supported by standardised ontology languages such as OWL, the accompanying tools for editing and maintaining ontologies also do not facilitate the representation of such knowledge.

We have thus chosen two approaches to *generate* ontologies with defeasible features for our evaluation. The first is a fully automated method to construct TBoxes (and DTBoxes) containing exceptions using random sampling techniques to construct subsumption statements from generated concept and role names. The second approach considers a principled manner in which to modify *existing* real world ontologies without defeasible features, to include such features. We report separately on the results for each of these datasets.

The algorithms that we evaluate are the ones presented in Chapter 4 namely Rational Closure, Lexicographic Closure, Basic Relevant Closure, Minimal Relevant Closure and Lexicographically Relevant Closure.

6.1 Artificial Data

We start off by reporting on the results conducted on artificially synthesised ontologies. While this approach is susceptible to biases in the generation strategy, it has been agreed upon to be a sensible preliminary methodology to obtain data for evaluation [30, p756-757], since there are no naturally occurring ontologies with explicitly modelled defeasible features.

6.1.1 Data Generation Model

Before we detail the actual method we use to generate our artificial ontologies and suitable entailment queries, we need to discuss some requirements that we have of the resulting data. In developing our methodology for generating synthetic ontologies we focus on two broad categories of parametrisation for

influencing the generation of an ontology: *global* and *local* parameters. Global parameters are those that pertain to the overall metrics of the ontology such as the number of axioms, classes, roles etc., while local parameters consider the structure of individual axioms (and class expressions) in the ontology. The latter parameters include factors such as nesting depth of expressions and the length of conjunctions and disjunctions.

We first discuss the global and local parameters chosen for generating *ontologies* and thereafter we consider the generation of axioms to be used as *entailment queries* in our evaluation. In the following subsection, we begin with the global parameters for ontologies and then move to the local parameters, concluding with a basic flow chart and overview of the ontology generation process.

Global Parameters

Firstly, one of the main goals of our evaluation is to obtain a preliminary indication of how "hard" defeasible reasoning would be in *real world* ontology development settings.

Therein lies a significant problem because we are not really in a position to accurately predict what these ontologies would look like when defeasible technologies become more widely adopted. One of the fundamental considerations, which we do not know a priori, is what percentage of the subsumptions in a real world ontology would users make defeasible?

Anecdotal evidence and our rudimentary lexical analysis of real world ontologies (see Chapter 3) suggests that the proportion of defeasible vs. strict information in real world ontologies would likely be lower than 25%. However there are various factors which could render this figure unreliable. For example, ontology engineers may have learned over time to avoid representing defeasible information in the meta-data of their ontologies because standardised ontology languages such as OWL and accompanying editing tools do not support the expression of defeasibility. In fact, the Marine Top Level Ontology [194] is an example of an application ontology that abides by the

monotonicity property as a matter of design practice [24, Page 9].
A positive aspect of generating our data is that we do not have to select a single value for the proportion of defeasible statements. We can actually consider a range of possible values. Therefore, our first decision for the artificial data is that we consider ontologies with varying ratios of defeasible to strict axioms and bin these ontologies into categories. We consider 10 categories for our evaluation. Each category represents a different *percentage defeasibility* (ratio of the number of defeasible vs. number of strict subsumption statements in the ontology) in increments of 10 from 10 to 100. We point out that the zero percent case of defeasibility means that defeasible reasoning is no longer applicable and therefore the case is omitted from our evaluation. For simplicity we only generate subsumption statements (either defeasible or strict) in our ontologies. That is, we do not generate syntactic sugar statements such as equivalence and disjointness statements. There is evidence to confirm that the number of subsumption statements far outweigh other forms of axioms in real world ontologies [97]. In any case, all of these latter axioms can be rewritten as subsumption statements.

Apart from the proportion of defeasible statements, we conjecture that it seems reasonable to assume that the remaining structure of real world defeasible ontologies might be very similar to that of existing non-defeasible real world ontologies. Therefore, in order to inform the parameterisation of our ontology generation method, it seems prudent to analyse some non-defeasible real world data to gather some metrics to use in our strategy. We use data from the recently established Manchester OWL Repository (MOWLRep) [135] for this purpose. The main motivation behind the establishment of the repository was to address biases in OWL empirical research where experiments are performed on cherry picked data or data lacking sufficient variety. The goal is to provide a platform for sharing high quality data with emphasis on variety for OWL empirical evaluations. We further motivate the use of MOWLRep for our investigations in Section 6.2.

Returning to our current concerns with generating artificial ontologies,

another factor to decide is the *size* of the ontologies to generate. Again, we can consider a range here. However, the emphasis with this first investigation is to give a very general sense of how defeasible reasoning would perform with ontologies of "reasonable" size. I.e., with "non-toy" examples. Research in classical DL reasoning optimisation is still grappling with the problem of reasoning with large-scale ontologies [89, 134], what then to speak of our defeasible reasoning algorithms which have to perform more work over the underlying classical reasoning steps? Therefore, for our purposes of gaining a preliminary insight into the performance of defeasible reasoning, we argue that it is not yet necessary to tackle large-scale ontologies in depth[1].

Although we do not generate large-scale ontologies we would still like to be somewhat representative of real world, non-defeasible ontologies in terms of the ontology sizes that we consider. Performing an analysis of the ontologies in MOWLRep, we found that the median ontology sizes in this dataset were around $3,800$ axioms (including non-TBox axioms). When restricting attention to TBox axioms the median ontology size obtained was $2,200$ axioms. Therefore, we choose to generate ontologies whose maximum sizes are capped at a figure within this range. In our resulting data the maximum ontology sizes we generated were approximately $3,500$ axioms. Such sizes are representative of numerous real world application ontologies in corpora outside MOWLRep as well (such as the SWEET corpus [157]).

In each percentage defeasibility category we would also like to have a minimum size for the generated ontologies. From our practical experience working with *application* OWL ontologies (i.e., those not built purely for demonstrational or educational purposes), the minimum sizes we have en-

[1]It is notable that the size of the ontologies in our dataset cannot be considered large-scale in comparison with some bio-medical ontologies such as those stored in the NCBO BioPortal corpus [196]. For example, the National Cancer Institute (NCI) thesaurus [181] appears in this corpus and has versions which contain more than $110,000$ axioms. At the same time, the concept hierarchies of most of these large bio-medical ontologies are rather shallow, making them less interesting from the standpoint of reasoning complexity. These ontologies also generally do not make use of all the expressive features available in $\mathcal{ALC}$.

countered of such ontologies have ranged between $150 - 250$ axioms. A good example of a small ontology (roughly 150 axioms) that is used for semantic web applications is the Friend of a Friend (FOAF) vocabulary[2]. We therefore choose 150 axioms as our lower bound for ontology size in each percentage defeasibility category. Recall that, even though we do not generate ontologies of large-scale size, one of our goals is still to present harder (yet not pathological) cases for our reasoner. In terms of isolating what makes reasoning hard for contemporary DL (and OWL) reasoners, there has been work done in prediction of classical OWL reasoner performance which suggests that the overwhelmingly dominant indicator of reasoning performance is ontology size [171]. Since our defeasible reasoner is built upon classical reasoning steps, it stands to reason that ontology size would be the primary indicator of performance in our context as well. Therefore, there is no reason to generate ontologies of smaller size than 150 axioms because there is no evidence to suggest that these cases would be harder for our algorithms.

In summary we generated 35 ontologies in each percentage defeasibility category, varying *uniformly* in size between roughly 150 and $3,500$ axioms. We argue that the number 35 is appropriate to give us a good spread of ontology sizes between 150 and $3,500$. The DL $\mathcal{ALC}$ is used to generate each ontology because the theoretical foundation of our algorithms has been explicitly investigated in the context of $\mathcal{ALC}$ (although, in principle, the results are applicable to a wide class of DLs up to the expressivity of $\mathcal{SHIQ}$). In terms of concrete syntax and format we express the generated ontologies using OWL with OWL/XML syntax. Since DLs are the logical underpinning of OWL, using data in the OWL format preserves the relevance of this evaluation. The main reason for using the OWL format, as opposed to a DL format, is the far superior availability of ontology data and tool support.

Our dataset thus consists of a total of 350 $\mathcal{ALC}$ ontologies with no ABoxes (our algorithms are specialised for TBoxes only at this stage). In order to represent defeasible subsumption in OWL ontologies (it is not included in the

[2] `foaf-project.org`

OWL specification) we "mark" relevant classical subsumption statements in the ontology as defeasible using meta-data constructs in OWL called *OWL annotations*[3]. Such constructs can be associated with specified OWL axioms and, using OWL processing tools such as the Java-based OWL API [95], one can programmatically identify the defeasible axioms in an OWL ontology.

Another global parameter which we consider for ontology generation is called *ontology signature size*. An ontology signature is the set of concept and role names mentioned explicitly in the ontology. We therefore have to consider the number of class names and role names to generate per ontology (relative to the number of axioms we wish to generate per ontology). Again consulting our analysis of MOWLRep we found that the number of concept names (respectively role names) per ontology were roughly 40% (respectively 1.5%) of the number of axioms in the ontology. Therefore these values are used in our ontology generation procedure.

The last global parameter that we consider for ontology generation is what we call *DL constructor distribution*. This is basically the proportion of axioms in the ontology which contain a particular DL construct ($\mathcal{ALC}$ construct in our case). That is, for each of the main $\mathcal{ALC}$ concept constructors: negation, disjunction, conjunction, existential and universal role restrictions, we are interested in the percentage of axioms in an ontology that contain each construct (whenever the ontology actually does contain the construct). This is the core variable or local parameter for our ontology generation methodology and, when examining the metrics of MOWLRep ontologies, we found the average values 6.2%, 26.6%, 21.1%, 4.3%, 14% and median values 1.5%, 17.8%, 11.1%, 2.2%, 4.3%, for negation, existential role restrictions, conjunction, disjunction and universal role restrictions, respectively.

Local Parameters

The structure of individual axioms in our generated ontologies can be influenced by many parameters. We reiterate that we simplify our task by

[3] `w3.org/TR/owl2-syntax`, Section 10.2

only generating subsumption statements in our ontologies. A subsumption statement has a left hand side (LHS) class expression and a right hand side (RHS) class expression. The structure, then, of a subsumption statement is defined by the structure of its LHS and RHS class expressions. We focus on two main parameters influencing the construction of a class expression: *nesting depth* and *conjunction or disjunction length*.

Nesting depth refers to the number of sub-class expressions in a given class expression. For example, the class name A has a nesting depth of 1, the expression $\exists \mathsf{R}.\mathsf{A}$ has a nesting depth of 2 (consisting of A and $\exists \mathsf{R}.\mathsf{A}$) and the expression $\exists \mathsf{R}.(\mathsf{A} \sqcap \mathsf{B})$ has a nesting depth of 4 (consisting of A, B, $(\mathsf{A} \sqcap \mathsf{B})$ and $\exists \mathsf{R}.(\mathsf{A} \sqcap \mathsf{B})$). Syntactic analysis of ontologies in MOWLRep reveals that, on average, the nesting depth of class expressions in real world ontologies is just 1. That is, the majority of classes in real world ontologies are names.

However, even though the average nesting depth is just 1, we have encountered isolated cases in MOWLRep where this number reaches 188 (and even one ontology where it reaches $1,707$). However, the majority of these larger nesting depths occur in the larger ontologies in MOWLRep (which are much larger than the ontologies in our synthetic dataset), so we opt for a lower maximum nesting depth for our synthetic data. We omit the strange case of $1,707$ from consideration because it is a single occurrence in the $22,000$ ontologies of MOWLRep. The next highest nesting depth is 188 and the accompanying ontology sizes for such occurrences is in the order of tens of thousands of axioms, whereas we have decided that the ontologies of our dataset should have a maximum of $3,500$ axioms. Therefore, we choose to cap the maximum nesting depth at 19 (one tenth of 188) for our ontologies.

Conjunction and disjunction length refers to the number of conjunct classes or disjunct classes *in a particular level of nesting* for a given class expression. For example, the *top level* disjunction length of the class expression $\mathsf{A} \sqcup (\exists \mathsf{R}.\mathsf{B} \sqcup \mathsf{D}) \sqcup \mathsf{C}$ is 3 even though the sub-expression $(\exists \mathsf{R}.\mathsf{B} \sqcup \mathsf{D})$ has a further 2 disjuncts. When examining class expressions (that actually contain conjunction and disjunction) in the ontologies of MOWLRep, we find that

the average conjunction length is around 2 and the average disjunction length is around 2.5. Just like in the case of nesting depth, the maximum values encountered are much larger. We encountered a maximum conjunction length of 85 in MOWLRep and we therefore choose a maximum conjunction length of 9 for our data. The maximum disjunction length is 194 and the next highest is 143 but these two cases are the odd ones out in the data (an order of magnitude larger than the remainder of maximum values in MOWLRep). Therefore, we choose the next highest value of 63 (a maximum disjunction length of 6) for our synthetic data.

Ontology Generation

We feed our selected global and local parameter values into a basic ontology construction procedure. The procedure consists of four main phases (a flow chart of this process is depicted in Figure 6.1). We give a brief description of each phase here.

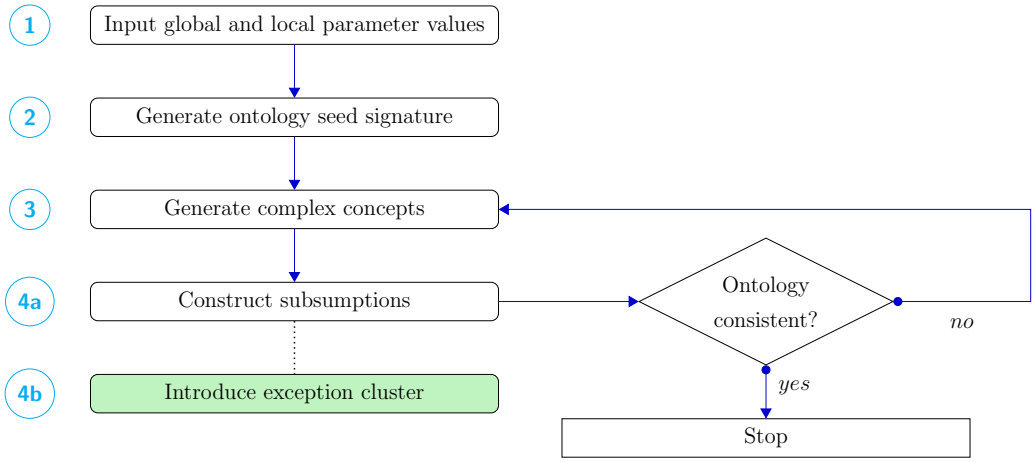


Figure 6.1: Basic flowchart of artificial ontology generation.

1. *Input global and local parameter values*: We first provide the input parameter values for the procedure. The main global parameters consist of the number of axioms to generate for the current ontology, the number of concept and role names to use in the construction of these axioms, the percentage of

the axioms to make defeasible, and the distribution of DL constructors across the axioms of the ontology. The local parameters are also given, namely, the maximum nesting depth and maximum conjunction or disjunction length of a class expression.

2. *Generate ontology seed signature*: The first main step of the procedure is to generate a set of concept and role names which would be the building blocks for constructing complex concepts and eventually subsumption statements in later steps of the process. If n (respectively m) is the number of concept names (respectively role names) to generate, then we generate concept names $\mathsf{A}_1, \ldots, \mathsf{A}_n$ and role names $\mathsf{R}_1, \ldots, \mathsf{R}_m$. We divide the concept names into two equally-sized disjoint sets representing LHS concept names and RHS concept names. The intention with this is that the concept names in each respective set are *predominantly* used as either LHS concepts or RHS concepts (and not both). We introduce a 98% chance that we do not use a concept name in the LHS set as a RHS concept name (and vice versa) in atomic subsumptions. As we shall see in later phases, this is necessary to ensure that we minimise the syntactic equivalences between concept names in our generated ontology.

3. *Generate complex concepts*: For each concept name in the LHS and RHS sets of Phase 2, we generate three complex concepts containing this concept name. The generated expressions are not divided into LHS and RHS expressions. This results in a total number of concepts that are sufficient to construct all the axioms in the ontology (recall that we are required to generate 2.5 times more axioms than concept names for the ontology). We use our provided maximum nesting depth and maximum conjunction and disjunction lengths to construct the complex concepts. Since the occurrence of the maximum values are very isolated in real world data, we also introduce a very small chance (around 0.01%) to generate expressions having values close to the provided maximums. We use the values obtained for DL constructor distribution to determine the chance of generating a class expression of a particular type. For example, the chance to generate a concept containing a universal restriction on a role would be around 4%. If we have to gener-

ate an existential or universal role restriction then we randomly select from the given set of role names in Phase 2. When we have to generate conjunctions or disjunctions, the conjuncts and disjuncts are randomly selected from the union of the LHS and RHS class name sets as well as newly introduced complex expressions.

4*a*. *Construct Subsumptions*: Analysis of real world ontologies such as those found in MOWLRep reveals that the majority of axioms describe relationships between names (class names). Therefore, we introduce a large chance (60%) to generate such axioms (a name is randomly selected from the LHS and RHS concept name sets from Phase 2, to define each axiom). In modern ontology editing systems, the perspective of ontologies is largely concept-centric (rather than axiom-centric). That is, editing is centred around specifying the subsumption relationship between concept names and concept expressions in the ontology (the subsumption relationship between a particular class name and other names or expressions is called its "definition"). More specifically, the definition of a concept name in editing tools is basically the enumeration of the set of concepts that are either a sub-concept of, super-concept of, equivalent with or disjoint with the given concept name. Therefore, in these tools the axioms that are defined using such interfaces are mostly of the form $\mathsf{A} \sqsubseteq \mathsf{B}$ or $\mathsf{A} \sqsubseteq \mathsf{C}$ (where A and B are concept names and C is a complex concept). As mentioned earlier we impose a 60% chance of generated axioms of the form $\mathsf{A} \sqsubseteq \mathsf{B}$. A 35% chance is assigned to generate axioms of the form $\mathsf{A} \sqsubseteq \mathsf{C}$. For the remaining 5% of cases we allow the generation of axioms of the form $\mathsf{C} \sqsubseteq \mathsf{D}$ (where both C and D are complex expressions). These latter types of axiom (also called general class axioms) are by far in the minority in real world ontologies. Constructed axioms are added three at a time to the ontology. The reason for this becomes clear in the optional Phase 4*b* which functions in tandem with Phase 4*a*.

4*b*. *Introduce Exception Cluster*: In order to present challenging reasoning cases to our defeasible reasoner, we have to ensure that there are exceptions in our generated ontologies. Our methodology thus far may or may

not "organically" introduce exceptions in the ontology, but, to make sure that there are exceptions we assign a small chance to introduce an *exception cluster* into the ontology. An exception cluster is a set of 3 axioms that represent classic defeasible inheritance example patterns. Take the running student example in this thesis: students generally don't pay taxes ($\mathsf{Student} \precsim \neg(\exists \mathsf{receives.TaxInvoice})$), but there are specific *types* of students that *do* generally pay taxes, i.e., employed students ($\mathsf{EmployedStudent} \sqsubseteq \mathsf{Student}$, $\mathsf{EmployedStudent} \precsim \exists \mathsf{receives.TaxInvoice}$). The general pattern is $C \precsim D$, $E \sqsubseteq C$ and $E \precsim \neg D$. A variant of this pattern is $C \precsim D$, $C \sqcap E \precsim \neg D$. We can also have exceptions-to-exceptions so we can extend both patterns to $C \precsim D$, $E \sqsubseteq C$, $E \precsim \neg D$, $F \sqsubseteq E$, $F \precsim D$ and $C \precsim D$, $C \sqcap E \precsim \neg D$, $C \sqcap E \sqcap F \precsim D$ respectively. The concepts C, D, E and F are randomly selected from the generated signature and complex expressions. We impose a 20% chance to introduce an exception cluster each time 3 axioms have been added to the ontology in Phase 4a.

Whenever we generate three axioms (that do not represent an exception cluster) in Phase 4a, we only add them as strict axioms to the ontology. Defeasible axioms are mainly introduced when we add exception clusters. Once we reach the target percentage defeasibility of the ontology we stop introducing exception clusters. If after generating the ontology the desired percentage defeasibility is still not met, we randomly select the remaining required number of axioms to be defeasible and "toggle" them to be such. To summarise the data generated using the above methodology, we give some relevant metrics of the ontologies in Figures 6.2a and 6.2b.

Entailment Query Generation

In addition to the ontologies we also randomly generated a set of defeasible subsumption statements (entailment queries) for each ontology using terms in their signatures (concept and role names in the ontology). The number of queries we generated per ontology was 1 percent of the ontology size. In other words, we chose to vary the number of generated queries proportionately

Classes			Roles			Ontology Size			Nesting Depth		Conjuncts		Disjuncts	
avg	median	max	avg	median	max	avg	median	max	avg	max	avg	max	avg	max
712	713	1,376	27	27	52	1,799	1,800	3,503	1	17	2	9	2	4

(a) Global and local metrics of generated ontologies.

Negation (%)		Conjunction (%)		Disjunction (%)		Existential (%)		Universal (%)	
avg	median	avg	median	avg	median	avg	median	avg	median
1.4	1.4	17.2	17	1.9	1.9	11.6	11.6	0.13	0.1

(b) Average $\mathcal{ALC}$ constructor distribution per generated ontology.

Figure 6.2: Relevant metrics and characteristics of the artificial ontologies.

according to the size of the ontology. This was to increase the chances of a generated query set being more "representative of", or "relevant to", the corresponding ontology as a whole (in terms of signature). All generated queries were stored to file together with their corresponding ontologies.

We claim that the value of 1 percent of ontology size (for the number of queries) is appropriate to give fairly representative average query times, while still guaranteeing termination of our experiments in reasonable time. For the LHS class expressions of the entailment queries we randomly selected from the *exceptional* LHS classes of defeasible subsumptions in the ontologies. This is to provide interesting and meaningful queries to our reasoner. If we ask queries with non-exceptional LHSs then defeasible reasoning reduces exactly to classical reasoning (only one classical entailment check is required) and the results would be less interesting for our purposes.

For the RHSs we would like to select expressions that are at least "relevant" to the LHS expression so that it makes sense to actually pose the queries to our reasoner. If the terms in the RHS expression are completely unrelated to the terms in the LHS expression then it is not meaningful to pose such queries to the defeasible reasoner. Therefore, we use the notion of modularisation (see Section 4.6) to achieve this. We extract a module (subset) of the ontology that is relevant to the terms in the LHS expression and then collect the terms in this module. The RHS expressions are then randomly generated from these terms. The generated subsumptions are all defeasible

because we are purely interested in the performance of "core" defeasible reasoning. Of course our algorithms themselves do support strict entailment queries but these queries follow from the ontology if and only if they follow via classical entailment from the strict axioms in the ontology. Therefore the performance of such reasoning tasks are irrelevant for our evaluation.

There are, obviously, a variety of ways to generate defeasible entailment queries. For example we could have used a strategy to *generate* the LHS expressions as well. We could have also randomly *selected* class expressions from the relevant module to stand as RHS expressions for our queries. However, we conjecture that our strategy represents a sensible first method for an investigation such as ours. Our test data (both ontologies and entailment queries) are available as a public download[4].

6.1.2 Experiment Setup

In this section, we give a description of our experimental conditions, the tasks that we execute, the important results we wish to report, and our hypotheses about how the algorithms would perform on the data.

Test Setup and Hardware

The first task was to generate the ranking for each of the ontologies in the dataset. We recorded the average time it took to generate a ranking (according to Procedure `ComputeRankingB` in Section 4.2) for the ontologies of each percentage defeasibility set. The rankings were all stored to file so that they would not have to be recomputed at a later stage.

The second set of tasks were to execute the generated set of entailment queries, using the Rational, Lexicographic, Basic, Minimal and Lexicographically Relevant Closures. We group our results for Rational and Lexicographic Closure together because they have a clear skeptical vs. credulous inference relationship. We group the results for Basic, Minimal and Lexicographically

[4] `krr.meraka.org.za/~kmoodley/ontologies/Synthetic.zip`

Relevant Closures together because these mechanisms all employ the use of justifications. We recorded the average time to compute an entailment using each of these procedures over the dataset.

In terms of optimisations, we use two main techniques described in Section 4.6 for ranking compilation and entailment checking. For ranking compilation the core optimisation is represented by Lemma 20 on Page 222. That is, if a LHS expression is not unsatisfiable (w.r.t. the classical counterpart of the ontology) it can never be exceptional (w.r.t. its defeasible form). Therefore, we only need to test exceptionality of class expressions that are unsatisfiable. For entailment checking, we can prune away axioms from the ranking that are irrelevant to the terms in the query being asked. Modularisation, as described in Section 4.6, is used for this purpose.

All experiments were executed on an Intel i7 Quad Core machine running Windows 10, with 8GB of memory allocated to the JVM (Java Virtual Machine). Java 1.7 is used with 3GB of memory allocated to the stack for running threads. For loading and analysing the ontologies of our dataset, we use version 3.5.4 of the popular and well-supported Java OWL API [95].

As we have shown in Chapter 4, our defeasible reasoning algorithms are built upon classical entailment checks. Thus, we would need to select an existing DL reasoning implementation to perform these classical entailment checks from within our defeasible reasoner. While running our evaluation with multiple implementations would have been interesting for comparison, such an investigation is not necessary to ascertain the price we pay for reasoning with defeasible (in addition to classical) subsumption. We therefore chose to utilise a single DL reasoner for our evaluation. In particular, we would ideally like to use the fastest and most robust implementation.

Consulting the latest results of the OWL Reasoner Evaluation Workshop[5], we identified the top three OWL 2 DL (expressive DL) reasoners for the standard reasoning tasks of: *classification*, *consistency checking* and *satisfiability testing* (in terms of performance and robustness). Robustness was measured

[5] `dl.kr.org/ore2014/results.html`

as the number of ontologies that were successfully processed in the allotted time. The top reasoners were Konclude[6] [185], HermiT[7] [79], MORe[8] [167], Chainsaw[9] [193], FaCT++[10] [192] and TrOWL[11] [190].

Modern DL reasoners are optimised for classification whereas various other reasoning tasks such as identifying unsatisfiable class names (incoherence) are usually performed by first classifying the ontology, and then "reading" the relevant information from the results.

Thus, we chose to focus on the reasoners which performed best in OWL 2 DL classification. These were respectively, Konclude, HermiT and MORe. Konclude, unfortunately, does not yet have a direct interface to the OWL API. Therefore, our choice was to select the next best reasoner - HermiT.

Hypotheses

The main insight we wish to gain from our evaluation is a general sense of the performance of defeasible reasoning (ranking compilation time and entailment query time) as well as clues as to where the major bottlenecks lie in this kind of reasoning. Our hypotheses are thus centered around these two insights. The first obvious hypothesis is that ranking compilation would be a much slower affair than entailment checking because there would be many more exceptionality checks in the former process.

The major bottleneck for ranking compilation should lie with the filtering out of hidden strict inclusions, and the main factor influencing the performance of this subprocess would be the number of iterations of the repeat loop (recursions) of Procedure `ComputeRankingB`. Note here that the major factor is not necessarily the *number* of hidden strict inclusions, but rather in the number of *times* hidden strict inclusions need to be transferred to the

[6] `derivo.de/produkte/konclude.html`
[7] `hermit-reasoner.com`
[8] `cs.ox.ac.uk/isg/tools/MORe`
[9] `chainsaw.sourceforge.net`
[10] `owl.man.ac.uk/factplusplus`
[11] `trowl.org`

TBox. Another factor that should influence perfomance is the length of the exception-to-exception chain for a particular ontology (the number of ranks in the ranking).

For Rational Closure we expect the main bottleneck to lie in computing the rank of C for a query $C \sqsubseteq D$. The higher the rank of C the more exceptionality checks are required to find C-compatibility. We also expect Rational Closure to be the best performing of the five entailment regimes because its worst case computational complexity is lower than the other regimes. The major bottleneck for Lexicographic Closure should be the identification of the LAC and the major factor influencing the performance of this subprocess is the size of the problematic rank. Recall that, in the worst case, the LAC has an exponential number of disjuncts (in the size of the problematic rank). Therefore, we expect the performance of Lexicographic Closure to drastically degrade when or if the problematic rank size increases drastically in the data.

For the Basic, Minimal and Lexicographically Relevant Closures we expect the majority of computation time to be taken by computation of justifications. The bottlenecks for these algorithms would then inherit the bottleneck of justification computation which is the construction of the hitting set tree (the major performance factors are the number of nodes in the tree and the number of justifications per entailment). We expect all the Relevant Closures to be slower than Rational Closure because they are in the double exponential time complexity class (worse than Rational Closure).

Because Lexicographically Relevant Closure combines mechanisms from Lexicographic Closure and Minimal Relevant Closure, it would inherit the bottlenecks from both. I.e., computation of justifications as well as computing the LAC. We have promoted the potential of Lexicographically Relevant Closure as a possible optimisation for Lexicographic Closure because of its potential to decrease the problematic rank size (by removing C-basis axioms from it). However, its success as an optimisation will then depend on justifications being much quicker to compute (than the LAC) and the C-basis size

being large relative to the problematic rank size (large portions of the problematic rank will then be pruned away making the LAC easier to compute). Finally, since Lexicographic Closure and the Basic, Minimal and Lexicographically Relevant Closures have the same worst case complexity we expect their performances to be more comparable with each other than with Rational Closure. We also expect Lexicographic Closure to be worst performing over the data (which means we anticipate many occurrences of large problematic rank sizes in the data). Basic and Minimal Relevant Closures would likely perform very similarly given their almost identical procedures. Although, we do anticipate that these procedures would perform significantly better than Lexicographic Closure over the data. Essentially we are anticipating that justification computation would be easier to perform than the computation of LACs in our data.

6.1.3 Ranking Compilation Results

It is important to note that we view the compilation of the ranking as an "offline" process prior to performing defeasible inference. That is, the ranking can, and should, be precompiled and stored to file whenever there is a stable version of the ontology. When reasoning needs to be conducted then the ranking is loaded and entailment queries can be posed with this ranking (the ranking should not be computed as part of every entailment query).

That being said, the ranking times we obtained for our data seem very reasonable considering that we have implemented only one optimisation for Procedure `ComputeRankingB`. As a point of reference, the average ranking times we observed in our data are comparable to the average times to compute all justifications for an entailment in the BioPortal corpus of ontologies [94, Figure 6.1, Page 99]. The percentile plot in Figure 6.3 gives a summary of the ranking times.

Percentile plots are chosen to represent the data because they give a good general picture of the performance for the majority of the data, and it also helps to reveal the outlier cases more clearly. For example, if we obtain

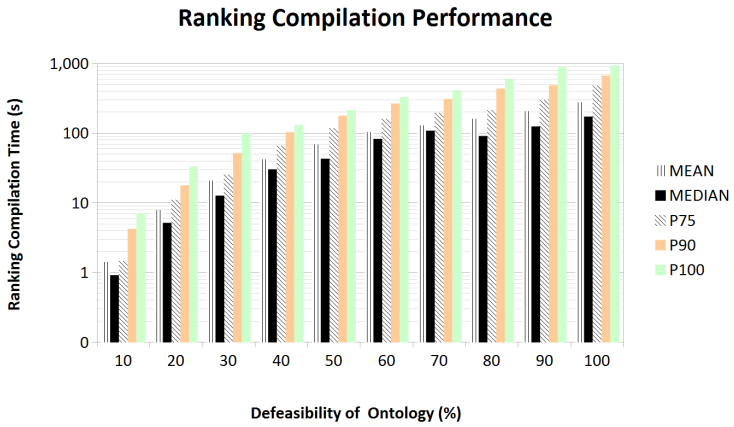
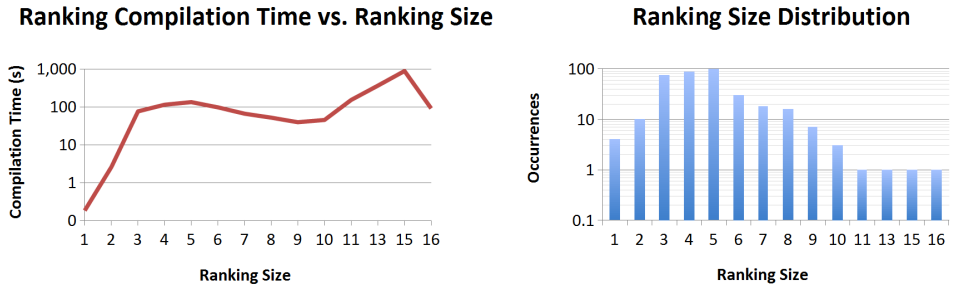


Figure 6.3: Average time to compute the ranking for the artificial ontologies.

a value of 5 seconds for the 90th percentile (P90) then it means that 90% of ontologies in the dataset could be ranked in 5 seconds or less. By this definition we note that the 50th percentile is actually the median value for a given dataset and P100 is the maximum value obtained.

Looking at the percentile plot of the ranking times in Figure 6.3, it seems that ranking compilation gets harder exponentially as the percentage defeasibility increases (the ontology sizes, including strict axioms, are roughly the same across the percentage defeasibility categories). This behaviour is to be expected since as percentage defeasibility increases, the proportion of defeasible axioms increases, and with this, the number of LHS class expressions (of defeasible subsumptions) that could potentially be unsatisfiable. Recall that for the ranking procedure, we have to perform an exceptionality check w.r.t. the ontology for each of these class expressions.

In addition, as we anticipated, the general trend is that ranking times increase with the number of ranks (also called the *ranking size* or the length of the exception-to-exception chain). Figure 6.4 illustrates this trend.
We observe that there is a dip in the curve between the ranking sizes of 5 and 10 and also between 15 and 16. The reason for these breaks in the trend is that these portions of the data coincide with brief declines in the percentage defeasibility of ontologies (percentage defeasibility is the other

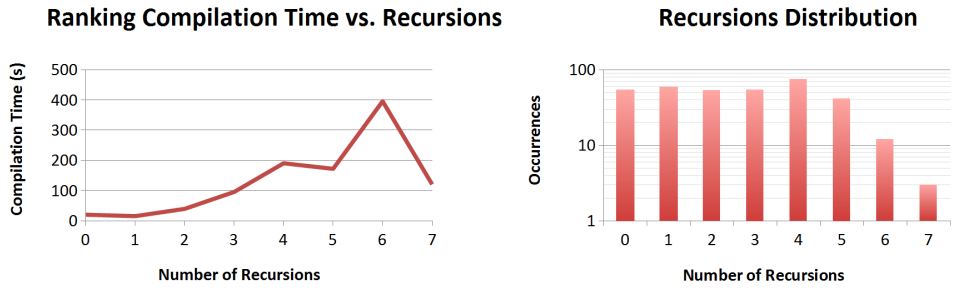


(a) Average ranking time versus the number of ranks in the ranking.

(b) Ranking sizes encountered together with their frequencies.

Figure 6.4: Influence of the ranking size on the ranking compilation performance. The Y-axis in Figure 6.4b denotes the number of ontologies in our data that have the indicated ranking size.

major factor influencing ranking compilation time). The other important factor is the number of times we have to recurse on the ranking procedure to filter out the hidden strict subsumptions. It is sensible to anticipate that when this recursion factor increases, our ranking times will also increase. This is confirmed in Figure 6.5.



(a) Ranking time vs recursion count.

(b) Recursion count frequencies.

Figure 6.5: Influence of the recursion counts on the ranking compilation performance. The Y-axis in Figure 6.5b denotes the number of ontologies in our data that have the indicated recursion count.

We have two dips in the curve of Figure 6.5a. One between recursion counts 4 and 5 and another between 6 and 7. The dips coincide with declines in

percentage defeasibility (from 72% to 64% and 83% to 67% respectively).

It must also be mentioned that the reliability of the curve shape in Figure 6.5a is greater between recursion counts 0 and 5. These are the most frequent counts found in the data (see Figure 6.5b) and thus the corresponding average values for the ranking time are more reliable in this range. The same can be said of the ranking size range between 3 and 5 for Figures 6.4a and 6.4b. We conclude this section with a summary of average metrics pertaining to the ranking compilation over the entire dataset (see Figure 6.6).

Defeasible Axioms	Ranking Size	Hidden Strict.	Rank 0 Size	Rank Size	Exception. Checks	Exceptional LHSs	Unsat. LHSs	Ranking Time (s)
991.2	4.8	500.6	312.9	98.5	2879.2	89.4	309.8	101.4

Figure 6.6: Average metrics pertaining to the ranking compilation per ontology. From left to right: number of defeasible axioms, ranking size, number of hidden strict subsumptions, size of the first rank (containing the non-exceptional defeasible axioms), number of axioms in a general rank, number of exceptionality checks to compute a ranking, number of exceptional LHS concepts of defeasible subsumptions, number of unsatisfiable LHS concepts of defeasible subsumptions and time to compute a ranking.

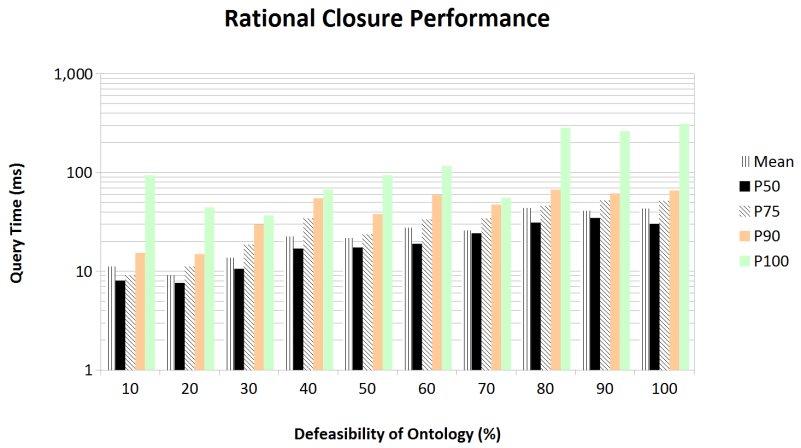
6.1.4 Entailment Checking Results

For entailment checking, we report first on the results for Rational and Lexicographic Closures. Recall that the main goal is to get a general idea of the practical performance of these algorithms as well as insights into where the main bottlenecks lie for these inference mechanisms.

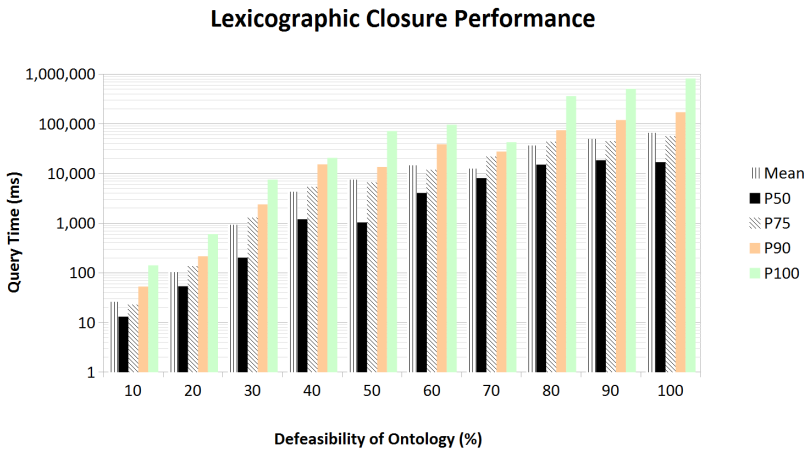
Rational and Lexicographic Closure

The performance results for both reasoning algorithms are encouraging. For Rational Closure the picture is especially bright even though the performance degrades by roughly one order of magnitude as the percentage defeasibility of the ontologies increase. For Lexicographic Closure the performance degrades more considerably (roughly $3 - 4$ orders of magnitude) as the percentage defeasibility increases. Nevertheless, for Lexicographic Closure, the median

values up to the 50% defeasibility case stay below the 1 second mark which is very reasonable performance for a largely unoptimised procedure. The overall results for the two algorithms are depicted in Figure 6.7.



(a) Average entailment checking times for Rational Closure.



(b) Average entailment checking times for Lexicographic Closure.

Figure 6.7: The average performance of Rational and Lexicographic Closure across our artificial dataset.

It is noteworthy that the average defeasible inference times using Rational Closure range between just 11ms and 43ms across the dataset (the maximum average time taken to compute an inference for any individual query in the

dataset, using Rational Closure, was 313ms). For Lexicographic Closure the performance is much slower, ranging between 25ms and 65s on average.

Although, looking at the median values for the latter, we find that the inference times drop considerably to between 13ms and 17s. This shows that there is wider variance in the inference times for Lexicographic Closure. That is, there are very hard cases and very easy cases also in the data. We will try to isolate where the hardness for these cases lie (for both Rational and Lexicographic Closure) later in this section.

For now we are in a position to give an answer to one of our questions at the start of this chapter (i.e., how much more intensive, on average, is defeasible reasoning than classical reasoning?). We plot the average number of classical entailment checks we use per defeasible entailment check for both Rational Closure and Lexicographic Closure. The results are illustrated by the graph in Figure 6.8.

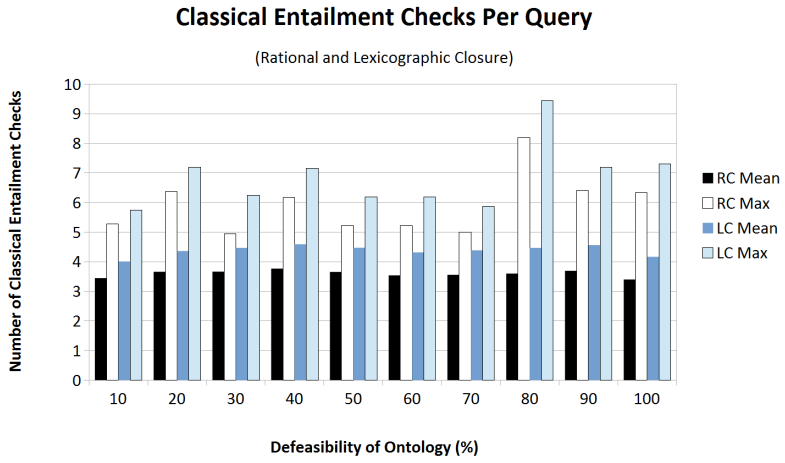


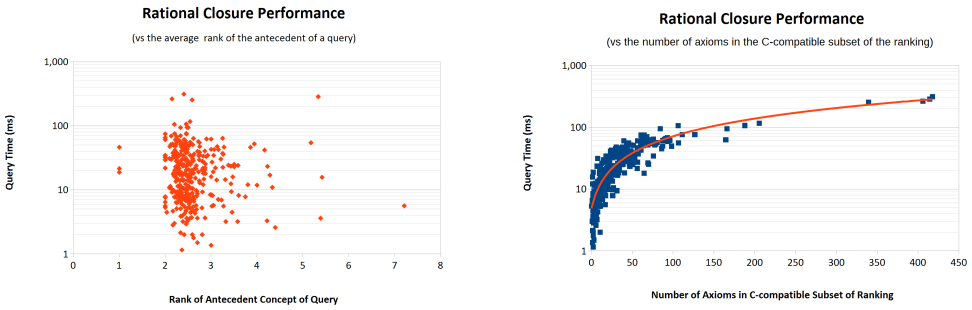
Figure 6.8: Average and maximum number of classical entailment checks per defeasible entailment check using the Rational and Lexicographic Closures. RC stands for Rational Closure and LC stands for Lexicographic Closure.

As we can see the graph depicts how many classical entailment checks it takes (on average) to compute a single defeasible entailment check. It is interesting that this value stays fairly consistent around the 3.5 mark for Rational Clo-

sure across the different percentage defeasibilities. Since classical entailment checks are the most computationally intensive components of our procedures for Rational Closure, we can make the generalisation that Rational Closure would likely take roughly 3.5 times as long as classical inference for $\mathcal{ALC}$ (for real world ontologies having sizes represented in our dataset).

Looking at the number of entailment checks for Lexicographic Closure we can see that they are not drastically more than Rational Closure. In fact, it takes roughly only one more entailment check to compute Lexicographic Closure (its average number of entailment checks stays consistently around 4.5). However, the very similar numbers of classical entailment checks belie the large discrepancy in performance between Rational and Lexicographic Closure. Therefore we have our first clear indication that the main performance factor Lexicographic Closure is not the number of classical entailment checks. Hence, we *cannot* make the generalisation that Lexicographic Closure takes roughly 4.5 times as long as classical entailment for $\mathcal{ALC}$.

Returning to Rational Closure, our hypothesis was that its major bottleneck would lie with the computation of the rank of the antecedent concept of the query being posed. Examining the graph depicted in Figure 6.9a, we find little evidence to support this hypothesis.



(a) Query time vs. the rank of the antecedent for RC.

(b) Query time vs. number of axioms in the C-compatible subset for RC.

Figure 6.9: Potential and actual main performance factors for Rational Closure.

Judging from the data points in the graph there seems to be no consistent

increase in query times as the average rank of the antecedent increases. What we can observe is that the average rank of query antecedents in our dataset lies predominantly between 2 and 3. Nevertheless, there is considerable variance in query time for this range of average antecedent ranks, from roughly 2ms to 150ms (Figure 6.9a depicts a logarithmic scale). Therefore, our hypothesis was false and there must be another variable which is contributing more to Rational Closure query time.

As mentioned earlier, the most computationally intensive components of the Rational Closure procedure are its classical entailment checks (each one is an EXPTIME-COMPLETE problem for $\mathcal{ALC}$). Figure 6.8 has illustrated that the number of classical entailment checks does not vary very much around 3.5 (not enough to warrant the large variance in query times between the antecedent ranks of 2 and 3). Therefore, it stands to reason that it is likely that the individual classical entailment checks themselves are taking longer than usual to compute for the hard cases. In other words, our hypothesis shifts to the suspicion that we are inheriting the main performance factor for classical entailment checking - which is well known to be ontology size.

This suspicion is actually confirmed by the correlation shown in Figure 6.9b. I.e., in this graph we can see that as the number of axioms in our C-compatible subset of the ranking increases, the query times also increase. The increase is quite dramatic until the C-compatible subset size is between 50 and 100 (the scale is logarithmic), thereafter the query times actually start to taper but still increase (although the number of occurrences of C-compatible subset sizes above this range also decreases dramatically). It is worth mentioning, then, that ontology size (specifically the number of defeasible axioms in the ontology) will always be a significant factor on performance for all our defeasible reasoning algorithms.

For the Lexicographic Closure we recall that our expectation was that the problematic rank size will be the major deciding factor for performance. We also anticipated "high" problematic rank sizes occurring quite frequently in our data. Figure 6.10 confirms the first expectation.

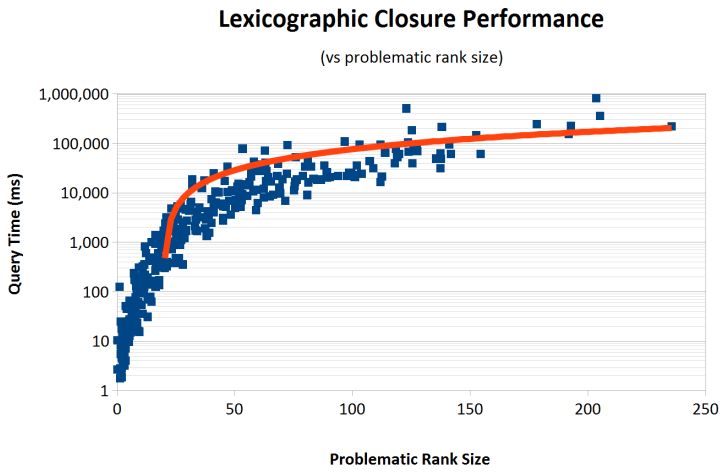


Figure 6.10: The main performance influencer for Lexicographic Closure in our dataset is problematic rank size.

Figure 6.10 also helps us to define what constitutes a "high" problematic rank size. The query times that fall below the $1,000$ms (1 second) mark are very acceptable for this relatively unoptimised implementation of Lexicographic Closure. We thus choose to assign the corresponding average problematic sizes for these cases as "low" problematic rank sizes. In our data this translates to values between 1 and 25. For queries that take between 1 and 100 seconds to terminate, we term the corresponding problematic sizes as "medium-sized" (the values range roughly between 25 and 125. Finally, for any queries that take longer than 100 seconds to terminate we consider the corresponding problematic rank sizes as "high". For our dataset this value is any number greater than roughly 125.

Therefore, it is sensible for those interested in keeping Lexicographic Closure performance very practical, to minimise the problematic rank size as far as possible. Outside the scope of optimisation techniques, an obvious design practice for defeasible ontologies to minimise problematic rank size is to reduce the number (or ratio) of defeasible axioms in the ontology.

In summary, we can extrapolate that the time to compute Lexicographic Closure (using our current algorithm) ranges between two times as long as

Rational Closure (in the 10% defeasibility case) to 3 orders of magnitude higher than those for Rational Closure (in the 100% defeasibility case). We can surmise, then, that Lexicographic Closure should range roughly between 7 and 3,500 times as long as classical entailment for defeasible ontologies of similar size to those in our dataset (recall that Rational Closure takes 3.5 times as long as classical entailment).

Basic, Minimal and Lexicographically Relevant Closure

For the Relevant Closures we notice, interestingly, that the performance for all three algorithms are almost identical. The performance discrepancy between Basic and Minimal Relevant Closure is just 0.01% and this is consistent with our hypothesis. Lexicographically Relevant Closure is around 0.18% faster than both Basic and Minimal Relevant Closure. We encountered isolated queries in the data that required inordinate reasoning times and thus, for pragmatic reasons, we had to impose a timeout of 1,000 seconds for each query to be able to terminate our experiments in reasonable time. We plot the general results for Basic Relevant Closure in Figure 6.11.

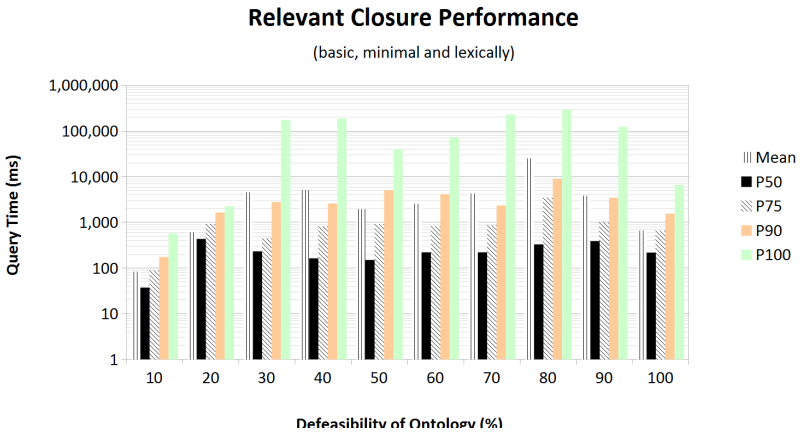


Figure 6.11: Overall performance of Basic Relevant Closure on the data. We omit the graphs for Minimal and Lexicographically Relevant Closures because their performance is almost identical with Basic Relevant Closure.

It is interesting to note that there is no marked increase in reasoning time

as the percentage defeasibility of ontologies increases. This is in contrast to what we witnessed with Rational and Lexicographic Closure. Since the computation of justifications is known to be an intensive procedure (relative to classical entailment checking), it is not surprising that the justification computation stage of the algorithms consumes over 99% of reasoning time.

Our results also indicate that the performance of computing justifications is not majorly affected by percentage defeasibility in our dataset. This is interesting and we conjecture that the reason for this is that the "relevant modules" for each query remain fairly consistent in size across the different ontology sizes. That is, justification computation makes use of modularisation to prune away axioms from the ontology that are irrelevant to the entailment. Therefore, it stands to reason that the size of these modules per entailment query stays consistent even when the ontology sizes change.

Looking at the maximum reasoning times obtained as compared to the 90th-percentile we can see there is a significant difference between the two. This means that there are a few isolated queries where the corresponding justifications were very hard to compute. We conjecture that the very high maximum reasoning times are responsible for elevating the mean times significantly above the median times as well.

Given the predominance of justification computation in the algorithms, the Relevant Closures should then, in principle, inherit the main performance factor for justification computation - the size of the hitting set tree (HST) generated. That is, the number of nodes in the HST tree. Horridge [94, Section 3.3.4] has explained that *justification overlap* (the degree to which justifications share common axioms) is a major factor in the practical performance of justification computation. That is, the more justifications actually share common axioms, the smaller the hitting set tree becomes and the faster it is to actually complete the tree construction - yielding all justifications.

Conversely, the less overlap there is between justifications, the more branches (or "repair paths") need to be generated from the combinations of unique axioms in the justifications, for the hitting set tree. In the worst

case, when there is no justification overlap whatsoever, the number of nodes in the hitting set tree grows to $2^{(n+1)} - 1$ where n is the number of justifications for the given entailment.

The hypothesis, then, that HST size will have a considerable effect on reasoning performance for the Relevant Closures holds true for our data. Figure 6.12 depicts a constant and linear increase in reasoning time as the HST size increase (both axes are represented using a logarithmic scale).

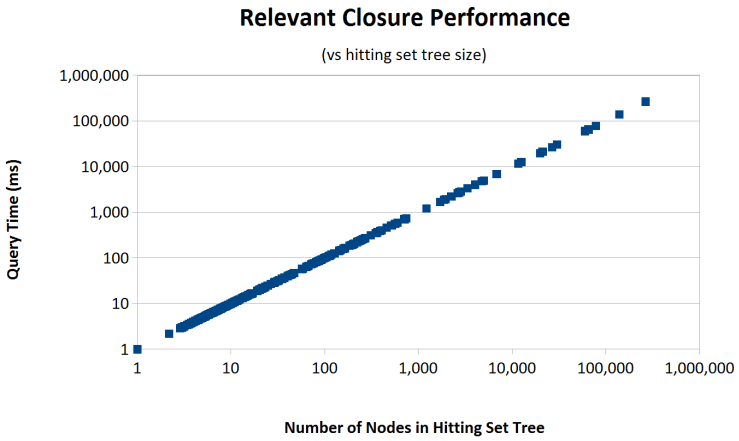


Figure 6.12: Influence of HST size on Relevant Closure performance.

As one can imagine, the justifications for the entailments in an ontology are very much determined by the modelling factors during ontology construction. The sheer variety in these modelling decisions causes variation in the makeup of justifications for entailments in the ontology. In fact, there is an entire subarea of ontology research which deals with the *justificatory structure* [22, 23] of ontologies. This area is concerned with the analytical study of various aspects of justifications in ontologies from the theoretical side, from the perspective of practical computation, as well as from a user comprehension (or cognitive) perspective. Therefore, in pursuit of considerable optimisations for the Relevant Closure algorithms, it is likely that one would have to gain decent insight into the justificatory structure of real world ontologies.
It must also be mentioned that we did not perform such a study in order to inform our parametrisation for our generated ontologies in this dataset. In

other words, it is not known how the justificatory structure of the ontologies in our dataset compares with that of real world ontologies. However, we have logged three key metrics for our data that we can compare with real world data: the average number of justifications computed, the average size of each justification and the HST size.

We also logged the average C-basis and minimal C-basis sizes encountered in our Basic and Minimal Relevant Closure algorithms. The values are shown in Figure 6.13.

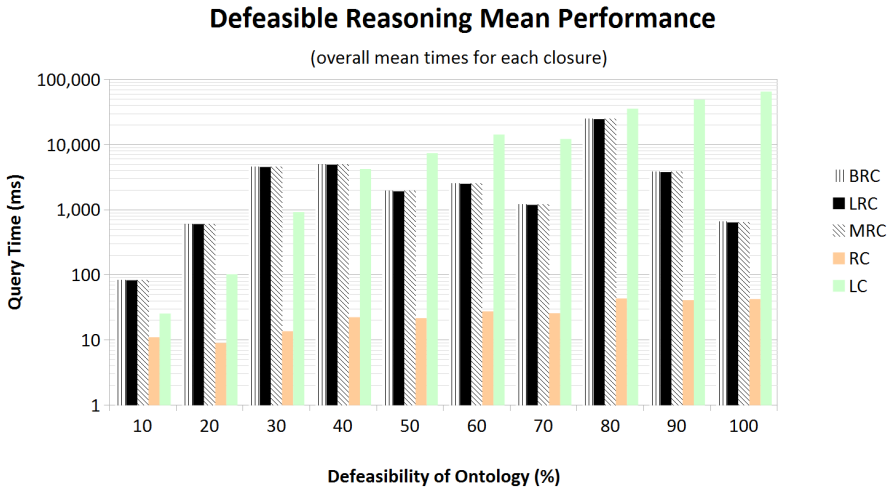
Case	CBasis			MinCBasis			Justifications			Just. Size			Hitting Set Tree Size		
	avg	median	max	avg	median	max	avg	median	max	avg	median	max	avg	median	max
Non-timeout	7	5	43	2	2	20	2	2	20	4	4	20	1,650	7	266,387
Timeout	71	69	139	17	16	34	46	45	126	14	13	30	1,045,536	1,000,007	2,000,005

Figure 6.13: Average (rounded off to the nearest whole number) metrics for computation of the Relevant Closure on the artificial data.

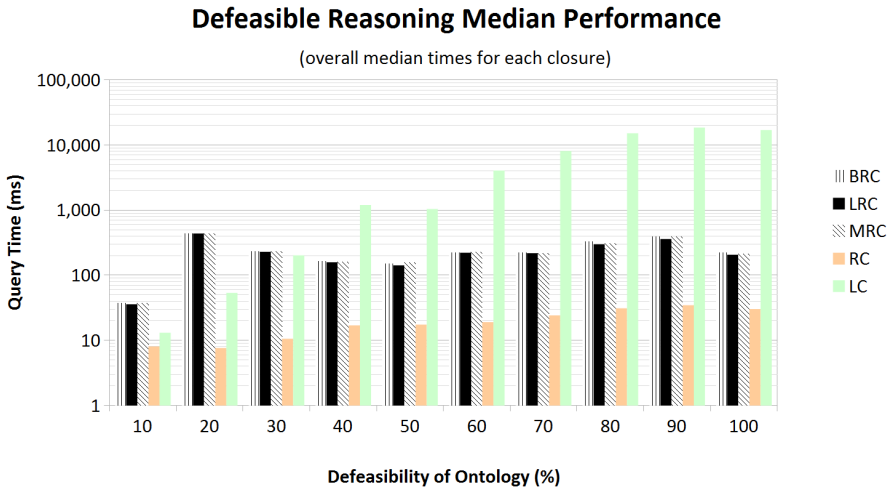
The values for the timed out cases in Figure 6.13 are those logged at the cut-off time i.e., at the $1,000$ second mark. It is interesting to note the very large difference between the average HST sizes of the non timed out cases and the timed out cases. The queries which could not be processed within $1,000$ seconds had to construct hitting set trees with $1,000,000$ nodes on average. For the cases that terminate we could construct HST sizes up to $266,387$ (the 99th-percentile was $35,334$) in the allotted $1,000$ seconds.

In terms of comparing the average performance of the Relevant Closures with the Rational and Lexicographic Closures, we obviously find that Rational Closure has, by far, the best performance on our data. Of course, this is consistent with our hypothesis considering also that Rational Closure generally removes more defeasible axioms from the ontology (and is thus inferentially weaker than the remaining algorithms). A performance comparison graph for the mean and median times of all algorithms is given in Figure 6.14.

The main extrapolations we can make from the graphs in Figure 6.14 are that Rational Closure is by far the reasoning paradigm with the strongest perfor-



(a) Mean times.



(b) Median times.

Figure 6.14: Mean and median times for all closures in the evaluation for the artificial data.

mance (around 2 to 3 orders of magnitude lower than the other algorithms). The Basic, Minimal and Lexicographically Relevant Closures are virtually identical in performance. In relation to Lexicographic Closure we find that the Relevant Closures are faster overall (on average over the entire dataset, Lexicographic Closure takes 4 times as long as the Relevant Closures).

However, this last fact can be misleading because we also notice that the mean performance of Lexicographic Closure is actually better than the Relevant Closures for the 10% to 40% defeasibility categories (LC takes approximately half the time of the Relevant Closures in this range). Thereafter, the performance of Relevant Closure overtakes that of Lexicographic Closure. The main reason is that from 10% to 40% defeasibility we have low problematic rank sizes (see Figure 6.15). From 50% to 100% the mean sizes increase slowly but the max sizes increase more dramatically which is perhaps to blame for the degradation in performance for LC.

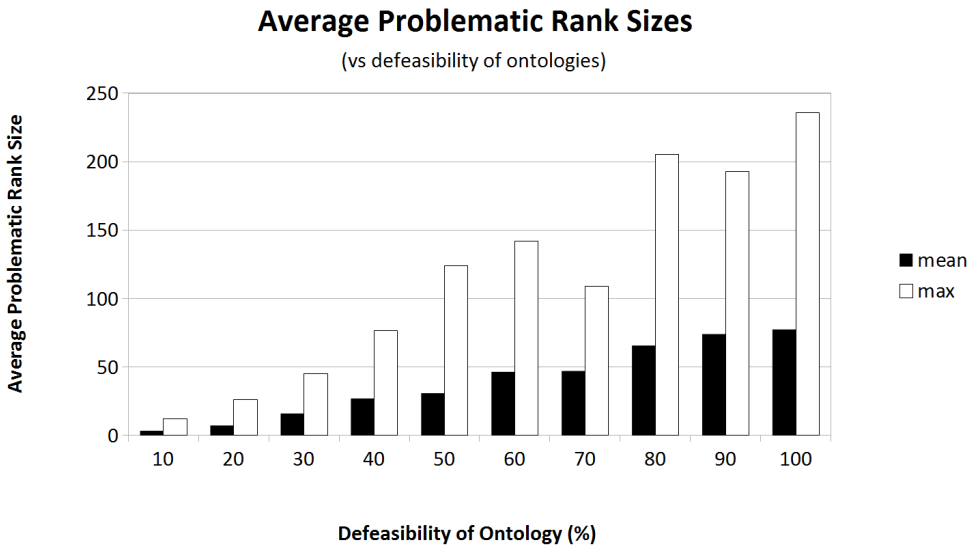


Figure 6.15: Average problematic rank sizes occurring in each percentage defeasibility category.

Another question we wished to answer at the start of this evaluation was if Lexicographically Relevant Closure is faster than (an optimisation for) Lexicographic Closure. The results that we obtained cannot definitively answer this question but they suggest that Lexicographically Relevant Closure excels (is faster than Lexicographic Closure) when the number of defeasible axioms (percentage defeasibility) increases above 50% for our dataset. Between 10% and 40% (for low problematic rank sizes) it is clear that Lexicographic Closure is without need of optimisation by way of computing justifications (as

Lexicographically Relevant Closure does).

6.1.5 Discussion

In summary for our artificial data, we believe that our results are sufficient grounds from which to claim that Rational Closure would be a viable defeasible reasoning formalism to be used in future real world ontologies containing defeasible features. What we mean here by viable is in terms of reasoning performance (reasoning times) using similar *reasoner interaction paradigms* to those used for classical DL reasoners in real world ontology editing tools.

A reasoner interaction paradigm is a manner in which users of modern ontology editing tools actually invoke the accompanying reasoners to derive knowledge from their ontologies. For example, classification is a very popular reasoning task in modern ontology editing software. The process of iteratively classifying an ontology to derive the main results for the users purposes is a typical reasoner interaction paradigm. In other words, we believe that Rational Closure would perform reasonably well in computing the *defeasible* subset or superset relationship between class names in a defeasible ontology (if or when a rudimentary non-naive algorithm for this is developed).

Another interaction paradigm would be for a user to supply as input an axiom, and for the reasoner to answer in the affirmative if the axiom is a logical entailment of the ontology, or in the negative if this is not the case. In essence this is the simplest task conceivable for defeasible reasoning and our evaluation results indicate that Rational Closure would perform very well for this kind of interaction.

Yet another interaction paradigm is embodied in the software plug-in for the ontology editor Protégé called the DLQueryTab[12]. In the DLQueryTab the user specifies a class expression and the main task of the plug-in is to invoke the reasoner to compute the set of all class names in the ontology that are either a subclass or superclass of the given expression. Again, considering that this task is related to the task of classification, we claim that Rational

[12] `protegewiki.stanford.edu/wiki/DLQueryTab`

Closure would also perform reasonably well at computing the defeasible subclasses or superclasses for the given expression.
Our main conclusion is that Rational Closure would give back inferences in times that can be considered "on-demand" using similar reasoner interaction paradigms to those mentioned above. Our confidence in this conjecture is based on the performance obtained in our evaluation of the artificial data (Rational Closure takes just 3.5 times as long as classical entailment) as well as the fact that we currently use only one optimisation for its computation (i.e., modularisation).

When we examine the results for Lexicographic Closure and the Relevant Closures we cannot make the same claims about their performance as for Rational Closure. Even though we have also used only modularisation as an optimisation for these algorithms, there is some doubt as to whether they would be useful (from a performance perspective) using similar reasoner interaction paradigms to those discussed above.

That is, whereas one might be willing to wait a few seconds or even up to a couple of minutes for the reasoner to determine if a specified axiom is defeasibly entailed by the ontology or not, whether it is feasible to wait 15 minutes (we even encountered some cases which take much longer) remains to be seen. However, if we consider the interaction with a defeasible reasoner to represent a completely different interaction paradigm to classical reasoning then perhaps one would be willing to wait that long. For example, waiting 2 or 3 minutes to compute *all* justifications for an entailment seems very reasonable on the whole (Horridge [94, Page 99] has shown that it sometimes takes close to 15 minutes for entailments in BioPortal ontologies).

Therefore, if defeasible reasoning were to be seen by users as embodying a different reasoner interaction paradigm (similar to the computation of justifications) then Lexicographic Closure and the Relevant Closures might still hold significance as useful forms of defeasible reasoning. It must also be reiterated that there is very little currently in the way of optimisations for all algorithms. Hence, the picture may be considerably brighter for these latter

algorithms when more sophisticated optimisation strategies are introduced.

6.2 Modified Real World Data

In this section, we take a step further than using purely synthetic ontologies. We describe a principled way of introducing defeasible subsumption into real-world ontologies. We then perform an evaluation of the performance of defeasible reasoning (analogous to that in Section 6.1) on the resulting data.

Previously, in terms of data for the evaluation of defeasible reasoning performance, the norm has been the use of automatically generated ontologies with defeasible features. The most notable attempt at a benchmark of synthetic defeasible ontologies is LoDEN[13]. LoDEN is however not applicable for our purposes because the focus in this benchmark is on *low complexity* DLs, whereas we are interested in ontologies that are at least of $\mathcal{ALC}$ expressivity.

Naturally, there are obvious shortcomings with artificially generated ontologies, such as possible biases in the generation methodology. However, there is no question of finding *representative* data because there are virtually no naturally occurring ontologies with intended defeasible features.

We instead choose a middle-ground approach, taking advantage of the rich set of (classical) OWL ontologies that we have on the Web in various repositories and corpora. Since DLs form the logical underpinning of OWL, this data is immediately applicable for our purposes. The basic idea of our approach is to "toggle" selected subsumptions in these ontologies to be *defeasible* subsumptions, thereby making these ontologies useful as data to evaluate our defeasible reasoning algorithms.

Of course, this is to be done with care to generate cases challenging for the reasoner. For example, we need to ensure cases where there are more than one rank in the ranking of the ontology (Procedure `ComputeRankingB`). Our method is given in Section 6.2.2, together with its strengths and weaknesses.

[13] `loden.fisica.unina.it`

6.2.1 Data Curation Methodology

For our initial data, we sample some classical OWL ontologies which we can later pass through our procedure for the introduction of defeasible features. The natural choice is to select the same data that is traditionally used to evaluate the performance of existing *classical* DL reasoners.

However, even in such a setting, there is no precise consensus on what data to use. The result is that data is generally curated manually by choosing "well-known" ontologies and corpora from which to sample, or arbitrarily selecting from the variety of respectable corpora on the web.

Choice of Corpora

While there are bona fide ontology benchmarks available such as LUBM [88] and its extension UOBM [129], it was pointed out that there are shortcomings in manual selection of ontologies and ontology corpora for evaluation of DL reasoning performance [133]. In particular, the main limitation with such selection procedures is that they result in datasets lacking sufficient *variety*.

Thus the results of evaluations can be heavily skewed or biased towards the particular benchmarks being used. The Manchester OWL Repository [134] is an effort to address this issue. The Repository is a framework for sharing ontology datasets for OWL empirical research. The current version of the repository contains three core datasets, namely versions of NCBO Bioportal[14] [148, 196], The Oxford Ontology Library (OOL)[15] and MOWLCorp[16] [133].

While Bioportal and OOL are already established ontology corpora that are actively used in DL reasoner evaluations, MOWLCorp is a recent gathering of ontologies through sophisticated web crawls and filtration techniques.

We obtain a recent snapshot of the Manchester OWL Repository as the base dataset for our evaluation. There are 344, 793 and 20, 996 ontologies in

[14] `bioportal.bioontology.org`

[15] `cs.ox.ac.uk/isg/ontologies`

[16] `mowlrepo.cs.manchester.ac.uk/datasets/mowlcorp`

the Bioportal, OOL and MOWLCorp corpora respectively.

Filtration Process and Choice of DL Reasoner

For loading and analysing the ontologies of our dataset, we use the popular and well-supported Java-based OWL API [95].

As we have shown in Chapter 4, our defeasible reasoning algorithms are built upon classical entailment checks. Just as in Section 6.1, we chose to utilise a single DL reasoner to perform these checks: the fastest and most robust implementation at the time - HermiT.

Given our choice of tools for manipulating and reasoning with the ontologies in our dataset, we filtered out the ontologies that could be loaded and parsed by the OWL API (each within an allotted 40 minutes). The remaining ontologies were then tested to determine if they were classifiable by HermiT within an additional 40 minutes each. Those ontologies which did not pass this test were also removed from the data.

In order to remove some of the cases which are very likely to be easy for our reasoner, we elected to remove ontologies with less than 100 logical axioms (ignoring annotations and other axioms carrying meta-information). This is justifiable because ontology size is proven to be an overwhelmingly dominant factor in reasoning performance [171] (see Section 6.1 for a more involved motivation of the number 100). Finally, we stripped the ontologies of ABox data because our defeasible reasoner is currently purely equipped with (D)TBox entailment procedures. This leaves us with 252, 440 and 2335 ontologies in Bioportal, OOL and MOWLCorp respectively.

6.2.2 Introducing Defeasibility into the Data

In this section, we describe a systematic technique to introduce defeasible subsumptions into the ontologies of our dataset, thereby making them amenable to defeasible reasoning evaluation.

Methodology

Our approach hinges upon an important relationship between *concept exceptionality* (Definition 14 on Page 91) and classical concept unsatisfiability. We rephrase this relationship (captured in Lemma 20 on Page 222) to be more intuitive for this setting:

Lemma 26 *If a concept C is exceptional w.r.t. a knowledge base $\langle \mathcal{T}, \mathcal{D} \rangle$ then C is unsatisfiable w.r.t. $\mathcal{T} \cup \mathcal{D}'$, where $\mathcal{D}'$ is the classical translation of $\mathcal{D}$.*

Lemma 26 on Page 294 states that if a class is exceptional in a defeasible ontology then it will necessarily be unsatisfiable in the classical translation of the ontology. This result is useful because we can use it to narrow down the search space for identifying exceptional classes in classical ontologies.

Taking the contrapositive of Lemma 26, we obtain the result that if a class is satisfiable w.r.t. a classical ontology then it is necessarily *not* exceptional w.r.t. any defeasible translation of the ontology. Therefore, we can eliminate ontologies from our dataset without LHS-classes of subsumptions that are unsatisfiable, because these could never become exceptional by turning classical subsumptions into defeasible ones.

The next definition is a generalisation of standard incoherence to axioms with complex left hand side (LHS) concepts:

Definition 63 (Classical LHS-coherence) *A classical TBox $\mathcal{T}$ is* LHS-coherent *if each $C \sqsubseteq D \in \mathcal{T}$ is s.t. $\mathcal{T} \not\models C \sqsubseteq \bot$. $\mathcal{T}$ is* LHS-incoherent *if it is not LHS-coherent.*

Eliminating all ontologies from our dataset that are LHS-coherent leaves us with 11, 46 and 77 ontologies in the Bioportal, OOL and MOWLCorp corpora respectively. Thus, in total we have 134 ontologies for our performance evaluation. The task is to "relax" some of the subsumptions of our ontologies to be defeasible. The obvious naïve approach to introducing defeasibility would be to convert *all* subsumptions to defeasible ones. Naturally, this is not likely to be the general approach of defeasible-ontology engineers in practice.

The other extreme would be to develop an approach to identify the *minimal* (for some defined notion of minimality) amount of defeasibility to introduce into the ontology in order to successfully "cater for all the exceptions". The latter approach would be interesting, and we are currently investigating such an approach; however, we propose that a reasonable approximation of such a procedure yields meaningful data for performance evaluation.

The approach that we discuss here is in the spirit of such an approximation. We illustrate the problem by means of an example:

Example 30 *Consider the following TBox* $\mathcal{T}$*:*

$$\left\{\begin{array}{lll}
1.\ \mathsf{Mechanic} & \sqsubseteq & \exists \mathsf{hasWorkshop}.\top, \\
2.\ \mathsf{Mechanic} & \sqsubseteq & \exists \mathsf{hasSpecialisation}.\top, \\
3.\ \mathsf{MobileMechanic} \sqcup \mathsf{GeneralMechanic} \sqcup \mathsf{CarMechanic} & \sqsubseteq & \mathsf{Mechanic}, \\
4.\ \mathsf{MobileMechanic} & \sqsubseteq & \neg\exists \mathsf{hasWorkshop}.\top, \\
5.\ \mathsf{MobileMechanic} \sqcap \neg\exists \mathsf{status}.\mathsf{OnStandBy} & \sqsubseteq & \exists \mathsf{hasWorkshop}.\top, \\
6.\ \mathsf{GeneralMechanic} & \sqsubseteq & \neg\exists \mathsf{hasSpecialisation}.\top, \\
7.\ \mathsf{CarMechanic} & \sqsubseteq & \exists \mathsf{hasSpecialisation}.\mathsf{Car}
\end{array}\right\}$$

MobileMechanic, GeneralMechanic and the class expression MobileMechanic $\sqcap$ $\neg\exists$status.OnStandBy are unsatisfiable w.r.t. $\mathcal{T}$. An intuitive analysis of $\mathcal{T}$ tells us that the ontology engineer probably intended to model that mechanics *usually* have a workshop (Mechanic $\sqsubseteq\!\!\!\sim \exists$hasWorkshop.$\top$) and *usually* specialise in certain types of equipment that they repair (Mechanic $\sqsubseteq\!\!\!\sim \exists$hasSpecialisation.$\top$).

This translation of Axioms 1 and 2 in Example 30 on Page 295, is a minimal and intuitive way to introduce defeasibility into $\mathcal{T}$, catering for exceptional types of mechanic - i.e., mobile and general mechanics.

However, we also have an exceptional type of *mobile* mechanic in $\mathcal{T}$ (an "exception-to-an-exception"). That is, mobile mechanics who are no longer "on standby" or "on call" (MobileMechanic $\sqcap$ $\neg\exists$status.OnStandBy). These mechanics would then be assigned a workshop for their repair tasks.

To cater for such mechanics we would have to relax Axiom 4 as well of Example 30 on Page 295 to express that mobile mechanics *usually* don't have a workshop (MobileMechanic $\sqsubseteq\!\!\!\sim \neg\exists$hasWorkshop.$\top$).

We now define a general *defeasible translation function* (DTF) for converting classical subsumptions to defeasible subsumptions in classical ontologies.

Definition 64 (DTF) *Let $\mathcal{T}$ be a set of classical subsumptions of the form $C \sqsubseteq D$, then $\mathscr{F} : \mathcal{T} \rightarrow \{C \mathrel{\tilde{\sqsubseteq}} D \mid C \sqsubseteq D \in \mathcal{T}\} \cup \mathcal{T}$ is a* DTF *for $\mathcal{T}$.*

We also have to formalise what we mean when a particular DTF "caters for all exceptions" in the TBox. We call such a function a *safe* DTF.

Definition 65 (Safe DTF) *Let $\mathcal{T}$ be a set of classical subsumptions, let $\mathscr{F}$ be a DTF for $\mathcal{T}$ and let $\mathscr{D}$ be the special DTF that translates* all *subsumptions in $\mathcal{T}$ to defeasible ones. Then, $\mathscr{F}$ is a* safe DTF *for $\mathcal{T}$ if C is totally exceptional w.r.t. $\mathscr{D}(\mathcal{T})$ if and only if C is totally exceptional w.r.t. $\mathscr{F}(\mathcal{T})$, for each $C \sqsubseteq D \in \mathcal{T}$.*

We try to define a safe DTF that places a small upper bound on the subset of axioms to relax using the well-known notion of justification [94].

If we compute the justifications for $\mathcal{T} \models \mathsf{MobileMechanic} \sqsubseteq \bot$ (the concise reasons for MobileMechanic being unsatisfiable and possibly exceptional) we obtain a single justification $\{1, 3, 4\}$. Relaxing these axioms would be sufficient for catering for mobile mechanics (in fact, it is only necessary to relax Axiom 1 as mentioned earlier). Similarly, we arrive at $\{2, 3, 6\}$ to cater for general mechanics and $\{4, 5\}$ for mobile mechanics no longer on call.

The basic idea is thus to take the union of the justifications for the unsatisfiable LHS-classes and relax these axioms to defeasible ones. We obtain that $\{1, 2, 3, 4, 5, 6\}$ should be relaxed in Example 30 on Page 295, which is admittedly a large proportion of our TBox. However, as we discover later in this section, the proportion is much smaller in practice on larger real-world ontologies.

However, while computing all justifications has been shown to be feasible in general on real-world ontologies, *black-box* (reasoner-independent) procedures are known to be exponential in the worst case [94]. To avoid this potential computational blowup, we obtain a small upper bound of the

(union of) justifications by extracting a *star locality based module* [170] for the ontology in question, w.r.t. the set of unsatisfiable LHS-classes.

A module of an ontology w.r.t. a signature (set of terms from the ontology) is a small subset of the ontology that preserves the meaning of the terms in the signature. We specifically choose star locality based modules because of two key properties: (i) they preserve all justifications in the ontology for all entailments (or axioms) that can be constructed with the given signature (*depleting* property [169, Section 3]), and (ii) they are smaller in size relative to other modules which have the depleting property. The pseudocode of our method is given in Procedure `RelaxSubsumption`.

Procedure `RelaxSubsumption(`$\mathcal{O}$`, `$\mathcal{C}$`)`

Input: LHS-incoherent TBox $\mathcal{O}$, $\mathcal{C} = \{C \mid (C \sqsubseteq D \in \mathcal{O}$ for some $D) \wedge (\mathcal{O} \models C \sqsubseteq \bot)\}$

Output: Defeasible ontology $\langle \mathcal{T}, \mathcal{D} \rangle$

1 $\mathcal{T} := \emptyset$; $\mathcal{D} := \emptyset$;

2 $\mathcal{M} := extractStarModule(\mathcal{O}, sig(\mathcal{C}))$; $\mathcal{T} := \mathcal{O} \backslash \mathcal{M}$;

3 **foreach** $X \sqsubseteq Y \in \mathcal{M}$ **do**

4 $\quad \mathcal{D} := \mathcal{D} \cup \{X \sqsubseteq Y\}$;

5 **return** $\langle \mathcal{T}, \mathcal{D} \rangle$;

Line 2 of Procedure `RelaxSubsumption` is responsible for extracting a hopefully small set of axioms from the ontology which preserves the meaning of the set of terms in $\mathcal{C}$ (the function $sig(\mathcal{C})$ extracts the signature or set of class and role terms in the set $\mathcal{C}$). Finally, it can be shown that our procedure defines a safe DTF for knowledge bases.

Theorem 6 (Safety of our DTF) *Let $\mathscr{F}$ be the DTF defined by Procedure* `RelaxSubsumption`*, and $\mathcal{O}$ a classical TBox. Then $\mathscr{F}$ is a safe DTF for $\mathcal{O}$.*

Proof: By Definition 65 on Page 296, we have to show that, for any concept C, C is totally exceptional w.r.t. $\mathscr{D}(\mathcal{O})$ if and only if C is totally exceptional

w.r.t. $\mathscr{F}(\mathcal{O})$ (where $\mathscr{D}$ is the special DTF which translates all subsumptions to defeasible ones).

" $\Longrightarrow$ ": Suppose that, for some $C \sqsubseteq D \in \mathcal{O}$, C is totally exceptional w.r.t. $\mathscr{D}(\mathcal{O})$ but C is *not* totally exceptional w.r.t. $\mathscr{F}(\mathcal{O})$. We show that this leads to a contradiction. From our supposition that C is not totally exceptional w.r.t. $\mathscr{F}(\mathcal{O})$ we have two cases: either C is not exceptional at all w.r.t. $\mathscr{F}(\mathcal{O})$ or C is *normally* exceptional w.r.t. $\mathscr{F}(\mathcal{O})$.

Case 1*:* C is not exceptional w.r.t. $\mathscr{F}(\mathcal{O})$. From our supposition that C is totally exceptional w.r.t. $\mathscr{D}(\mathcal{O})$ we can infer that $\mathcal{O} \models C \sqsubseteq \bot$ from Lemma 26 on Page 294. Let $\mathcal{J}_1, \ldots, \mathcal{J}_n$ be the justifications for $\mathcal{O} \models C \sqsubseteq \bot$. Because we know that C is totally exceptional w.r.t. $\mathscr{D}(\mathcal{O})$ it must be the case that for at least one $1 \leq i \leq n$, C is totally exceptional w.r.t. $\mathscr{D}(\mathcal{J}_i)$. We can easily see from the depleting property of star locality based module that $\mathscr{D}(\mathcal{J}_i) \subseteq \mathscr{F}(\mathcal{O})$. Therefore C is totally exceptional w.r.t. $\mathscr{F}(\mathcal{O})$ because there is a justification for this in $\mathscr{F}(\mathcal{O})$. This is a contradiction and therefore it cannot be the case that C is not exceptional w.r.t. $\mathscr{F}(\mathcal{O})$.

Case 2*:* C is normally exceptional w.r.t. $\mathscr{F}(\mathcal{O})$. This is impossible because we have shown in Case 1 that there is a justification $\mathcal{J}_i$ for $\mathcal{O} \models C \sqsubseteq \bot$ s.t. C is totally exceptional w.r.t. $\mathscr{D}(\mathcal{J}_i)$. Therefore $\mathscr{D}(\mathcal{J}_i) \subseteq \mathscr{F}(\mathcal{O})$ and it must be the case that C is totally exceptional w.r.t. $\mathscr{F}(\mathcal{O})$.

" $\Longleftarrow$ ": Suppose that, for some $C \sqsubseteq D \in \mathcal{O}$, C is totally exceptional w.r.t. $\mathscr{F}(\mathcal{O})$ but C is *not* totally exceptional w.r.t. $\mathscr{D}(\mathcal{O})$. We know there is a justification $\mathcal{J} \subseteq \mathscr{F}(\mathcal{O})$ s.t. C is totally exceptional w.r.t. $\mathcal{J}$ and $\mathcal{J}' \models C \sqsubseteq \bot$ where $\mathcal{J}'$ is the classical counterpart of $\mathcal{J}$ (Lemma 26 on Page 294). Assume there is a classical statement in $\mathcal{J}$. This is impossible because $\mathscr{F}$ is defined s.t. each $\alpha \in \mathscr{F}(\mathcal{O})$ where α' (classical counterpart of α) is part of a justification for the unsatisfiability of C, has to be part of the star locality-based module we consider. This means α' will be translated to defeasible in $\mathscr{F}(\mathcal{O})$. Therefore, it must be the case that $\mathscr{D}(\mathcal{J}) = \mathcal{J}$. Then there is a justification for C being totally exceptional w.r.t. $\mathscr{D}(\mathcal{O})$ because $\mathscr{D}(\mathcal{J}) \subseteq \mathscr{D}(\mathcal{O})$. Hence it must be that C *is* totally exceptional w.r.t. $\mathscr{D}(\mathcal{O})$. □

Discussion

There are two conflicting issues with the procedure we have presented for introducing defeasibility into OWL ontologies: (i) minimality of modification to the original ontology and (ii) the representative quality of the resulting defeasible ontology as something that might be built by an ontology engineer with access to defeasible features.

While (i) and (ii) would be a useful goals for a methodology automating the introduction of defeasible features into OWL ontologies, our approach does not yet meet such desiderata. It is clear that the minimal axioms to relax in Example 30 on Page 295 would be $\{1, 2, 4\}$, yet we relax $\{3, 5, 6\}$ as well.

The resulting ontology should also ideally resemble a naturally occurring ontology with defeasible features introduced where explicitly needed by the ontology engineer. For instance, in Example 30 on Page 295, it does not make sense (from an intuitive point of view) to relax $\mathsf{MobileMechanic} \sqsubseteq \mathsf{Mechanic}$ (all mobile mechanics are mechanics) to $\mathsf{MobileMechanic} \sqsubseteq\!\!\!\!\sim \mathsf{Mechanic}$ (*typical* mobile mechanics are mechanics). Such constraints should ideally remain strict.

Furthermore, a critical observation is that logical incoherence in classical ontologies may be caused by erroneous modelling. In ontology development tools, large emphasis has been placed on debugging incoherence by making modifications to the ontology to remove the "unwanted" entailments such as $C \sqsubseteq \bot$. This is likely to have prevented many developers publishing incoherent ontologies.

Given the above main shortcomings of our approach, we do not argue that ours is the ideal methodology. Rather, we hope that it serves as a stepping stone from purely synthetic approaches to investigate and develop more suitable methodologies. Our modified real world ontologies are available as a public download[17].

[17] `krr.meraka.org.za/~kmoodley/ontologies/NonSynthetic.zip`

6.2.3 Experiment Setup

Our setup, methodologies and design choices for the experimental evaluation can be summarised as follows:

Data Summary

The input data for our experiments are 134 LHS-incoherent ontologies (curated as described earlier in this section) from the Manchester OWL Repository. The ontologies are divided across three corpora: 11, 46 and 77 in Bioportal, OOL and MOWLCorp respectively. The DL expressivity distribution of the data ranges from variants of $\mathcal{ALC}$ all the way up to $\mathcal{SROIQ}$ [99]. There are 35 DL variants in total represented in the data. Figure 6.16 provides some average properties of the ontologies in our dataset.

Corpus	Classes			Roles			TBox size			RBox size			Nested Classes		Conjuncts		Disjuncts	
	avg	median	max	avg	median	max	avg	median	max	avg	median	max	avg	max	avg	max	avg	max
Bioportal	17,992	422	187,515	50	33	152	41,309	754	439,208	30	17	149	1	28	2	10	2.1	5
OOL	22,203	16,306	89,926	55	44	194	41,389	32,928	142,996	27	9	123	1	136	2	68	2.5	16
MOWLCorp	4,621	216	89,926	466	13.5	16,586	8,719	691	137,021	214	5	7,749	1	34	2	17	2.7	16

Figure 6.16: Ontology metrics for the LHS-incoherent cases in the dataset.

We also give an illustration in Figure 6.17 of how much defeasibility our methodology has introduced in to the curated ontologies. The average ratio of defeasible to strict axioms in each ontology is 6%, the median being 1%, the minimum ratio being 0.01% and the maximum being 98%.

It is very interesting to note that the percentage defeasibility of most ontologies in the data stay well below 10%. It would be interesting to see if our reasoning performance for the 10% defeasibility category of the artificial data (Section 6.1) generalises to the current data as well. There are a number of factors, though, which make the current data different from the artificial data. The main one in terms of performance is probably ontology size. In our current data we have far larger ontologies in general than the artificially generated ones.

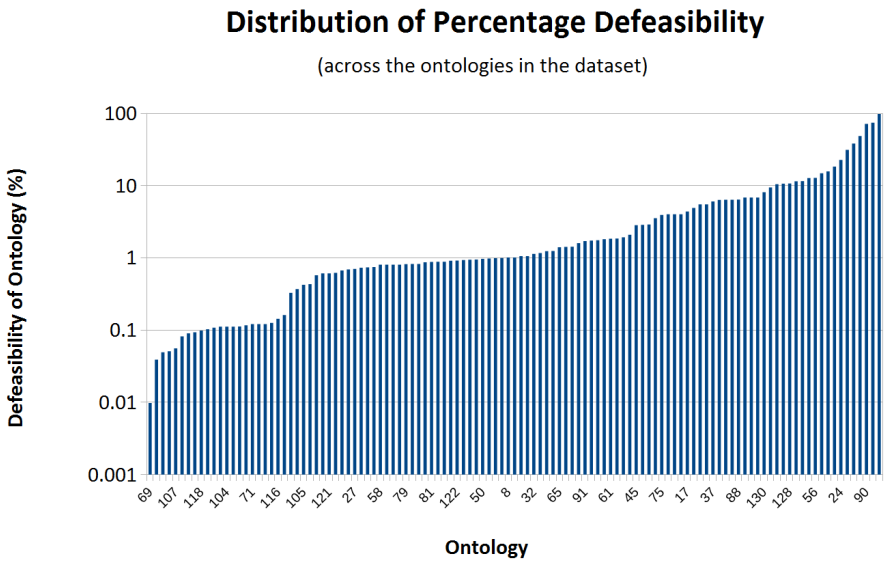


Figure 6.17: Percentage defeasibility distribution across the modified real world ontologies.

In terms of the $\mathcal{ALC}$ constructor distribution, Figure 6.18 shows that for the most part the distribution for the real world data closely matches that of the artificial data (see Figure 6.2b). The only discrepancy is with universal role restrictions which occur more frequently in the real world data than in our artificial data. While this is not ideal, we conjecture that overall this would not detract from the significance of the results for the artificial data.

Negation (%)		Conjunction (%)		Disjunction (%)		Existential (%)		Universal (%)	
avg	median	avg	median	avg	median	avg	median	avg	median
6.16	1.49	21.07	11.06	4.25	2.2	26.55	17.81	13.97	4.29

Figure 6.18: Average $\mathcal{ALC}$ constructor distribution across an ontology in our modified real world dataset.

In addition to the ontologies, we generated a set of entailment queries (defeasible subsumptions of the form $C \sqsubseteq D$) for each ontology to present to our defeasible reasoner. We follow a similar strategy to Section 6.1: for the C's we use the existing LHS-incoherent classes in the ontology. For the D's we take the $\bot$-syntactic locality module of the ontology w.r.t. to the signature of C (including $\bot$) and randomly generate RHS D's from the module signature.

Tasks

The first task is to precompile the exceptionality rankings of each ontology in the dataset. The rankings are then stored on file for later use in entailment testing. It is important to note that the computation of the ranking is considered as an offline, precompilation process for each stable version of an ontology. Such a task is not meant to be executed on-demand during defeasible entailment tests.

Lemma 26 on Page 294 is used as an optimisation in the ranking procedure. We only need to check exceptionality of $C \sqsubseteq D$'s where C is unsatisfiable w.r.t. the classical translation of $\langle \mathcal{T}, \mathcal{D} \rangle$ (see Lines 2 to 4 of Procedure `Exceptional` in Section 4.2).

The entailment tests are then performed on the precomputed rankings and the results of both tasks are recorded. I.e., we test if $C \sqsubseteq D$ is follows from the ontologies using the various closures presented in Chapter 4. We recorded the average time it took to compute the rankings, and to answer entailment questions. The same optimisations were used as for the artificial data evaluation.

Equipment

The evaluation was carried out on an Intel i7 Quad Core machine running Windows 10. 8GB of memory is allocated to the Java Virtual Machine (Java 1.7 with 3GB of memory allocated to the stack for running threads). HermiT is the underlying classical DL reasoning implementation.

Hypotheses

Our general predictions are analogous to those of the artificial data because the ontology metrics (apart from ontology size, classical subsumptions and percentage defeasibility distribution) of both datasets are somewhat similar. However, we expect the data to contain ranking sizes not more than roughly 5. That is, we expect there to be very few cases where there are

exceptions-to-exceptions (if any). The reason for this prediction is that ontology engineers would probably try to avoid introducing incoherences in their ontologies which correspond to exceptions.

6.2.4 Ranking Compilation Results

It must be pointed out that the presentation of our results for this data is going to be significantly different to that of the artificial data in Section 6.1. The reason is that we do not have the ontologies binned into neat categories w.r.t. percentage defeasibility.

The overall results for ranking compilation are quite promising. Figures 6.19 and 6.20 give an overview of results w.r.t. ranking compilation.

TBox size			DTBox size			LHS-incoherent classes			Totally exceptional LHS classes		Ranking size		Ranking Time (s)		
avg	median	max	avg	median	max	avg	median	max	avg	max	avg	max	avg	median	max
26,089	861	415,258	354	39	7,842	138	4.5	4,716	12	322	1.5	3	176.40	0.21	16,263

Figure 6.19: Ontology metrics and ranking compilation results for the modified real world data.

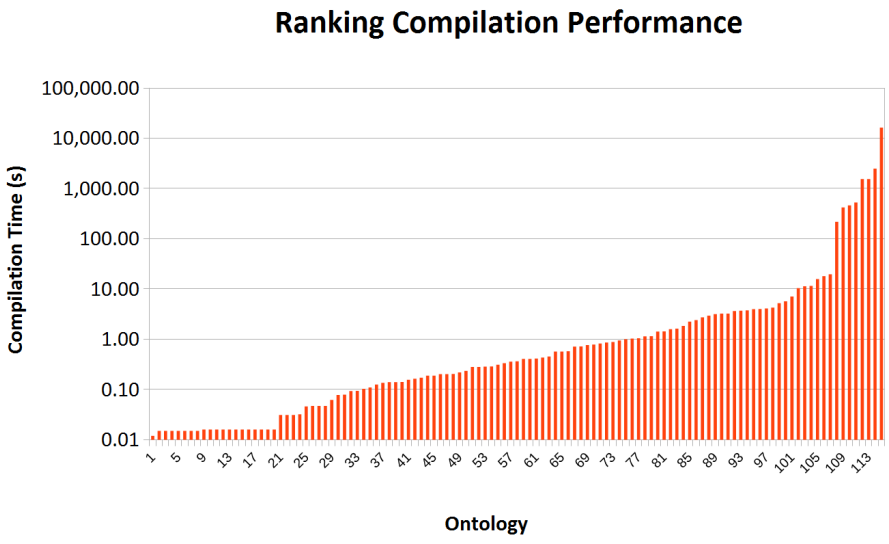


Figure 6.20: Ranking compilation time per modified real world ontology.

Examining the ranking times in Figures 6.19, we notice that on average over the entire dataset, it takes 3 minutes to rank a single ontology. However, the "median" column of the ranking times, shows us that the majority of rankings were computed in less than half a second. The most intensive ranking to compute was Ontology 134 which has 415,258 TBox axioms of which 6,010 are defeasible (it took roughly 4 hours to compute).

As expected, we have very little variance in ranking size (between 1 and 3), therefore we do not need to examine in detail the influence of ranking size on the compilation time. However, the most challenging cases in theory for our reasoner are the ones in which there are hidden strict inclusions in the DT-Box. Examining the number of recursions we have to perform over the data, we find that the need to recursively execute Procedure `ComputeRankingB` is much less frequent than the artificial data (see Figure 6.21).

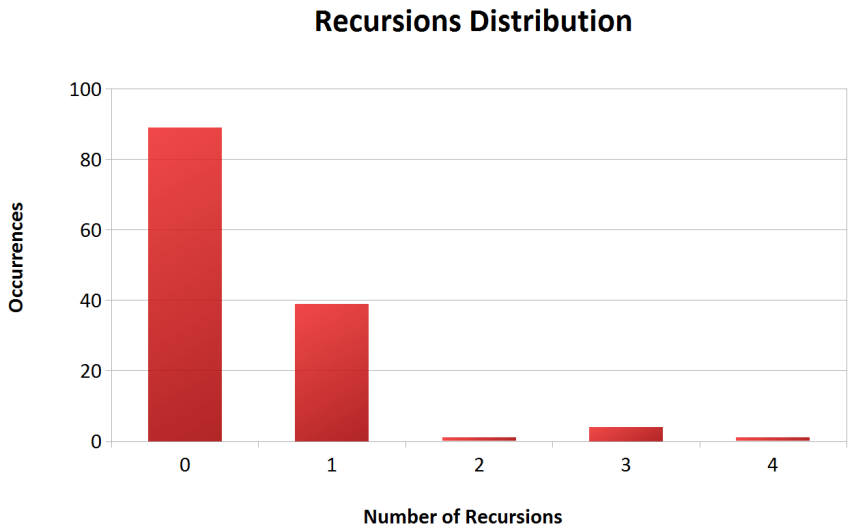


Figure 6.21: Number of recursions required to rank the modified real world ontologies.

Therefore, the number of recursions does not impact the hardness of ranking compilation considerably because average recursion depth is 1.5 with the maximum of just 4 (and very little occurrences of 2, 3 and 4 in the data). This is also confirmed by the fact that the average compilation time for the cases

with no recursions is by far the highest (255 seconds). Hardness, therefore, is mainly determined by other factors. Since a naive ranking compilation procedure has to check exceptionality of each defeasible axiom in the ontology, in most cases we expect the number of defeasible axioms to be the main contributor to hardness. However we also have an optimisation (Lemma 26 on Page 294) which says that we only need to check exceptionality of the defeasible axioms with unsatisfiable LHSs.

Therefore, it stands to reason that the number of unsatisfiable LHSs in the ontology would be the main contributor to hardness for our dataset. We plot the number of defeasible axioms in the ontology that have unsatisfiable LHSs against the ranking compilation time to reveal that this is indeed the case (see Figure 6.22).

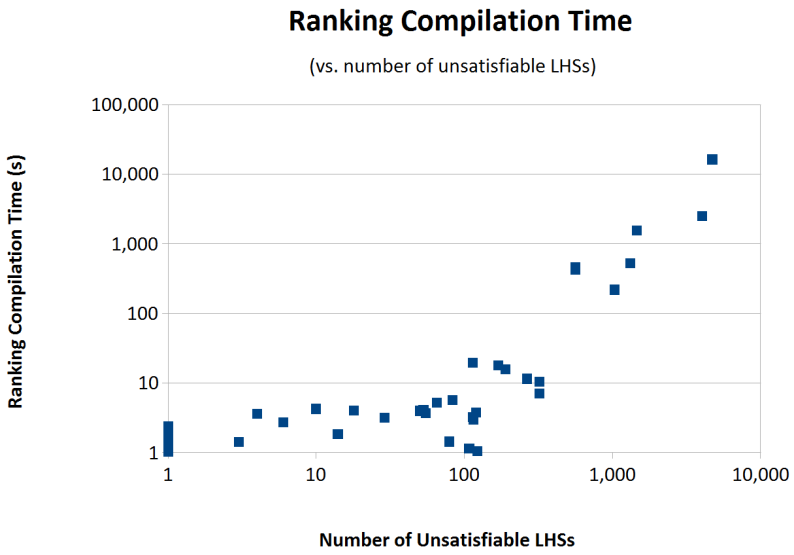


Figure 6.22: The performance of ranking compilation vs the number of defeasible axioms in the ontology that have unsatisfiable LHSs.

Both the X and Y axes are represented in logarithmic scale and we can see that from around 100 unsatisfiable LHSs the ranking times start to increase dramatically. In summary the compilation times for the modified real world data are, in general, comparable with those for our artificial data.

The average time to compile a ranking in the artificial data was around 100 seconds, whereas in our modified real world data this figure is around 176 seconds. However, we have much smaller percentage defeasibility ratios in the latter dataset than we do in the former. It would be interesting to see in the future whether real world defeasible ontologies would have similar percentage defeasibility ratios to those in our dataset.

An analysis of our algorithm, together with the results obtained in this evaluation, reveals that the number of unsatisfiable LHSs (and to a lesser extent the number of recursions) are the main contributors to hardness of ranking compilation.

It must also be mentioned that one should not ignore the number of strict axioms (i.e., overall ontology size with both defeasible and strict axioms) as a contributor to reasoning hardness. As we have repeatedly stressed, our algorithms are built upon classical entailment checks for which ontology size is the dominant indicator of hardness.

In fact, we notice that the average number of defeasible axioms in our real world dataset is only one third that of the artificial data, and yet we still obtained some cases in the real world data where compilation is harder than in the artificial data. We attribute most of this to the fact that ontology sizes are much larger on average in the real world dataset (see Figures 6.19 and 6.16 for a comparison).

To conclude the ranking compilation analysis, we give some average metrics of this part of the evaluation in Figure 6.23.

Defeasible Axioms	Ranking Size	Hidden Strict.	Rank 0 Size	Rank Size	Exception. Checks	Exceptional LHSs	Unsat. LHSs	Ranking Time (s)
315.39	1.5	11.9	264.18	151.06	227.79	39.33	138.27	176.38

Figure 6.23: Average metrics obtained during the evaluation of ranking compilation performance for the modified real world data.

6.2.5 Entailment Checking Results

As we did in Section 6.1, we report first on the results for Rational and Lexicographic Closure and then on the results for the Relevant Closures. We also give a brief comparison and discussion of the results for both groups of closures at the end of this section.

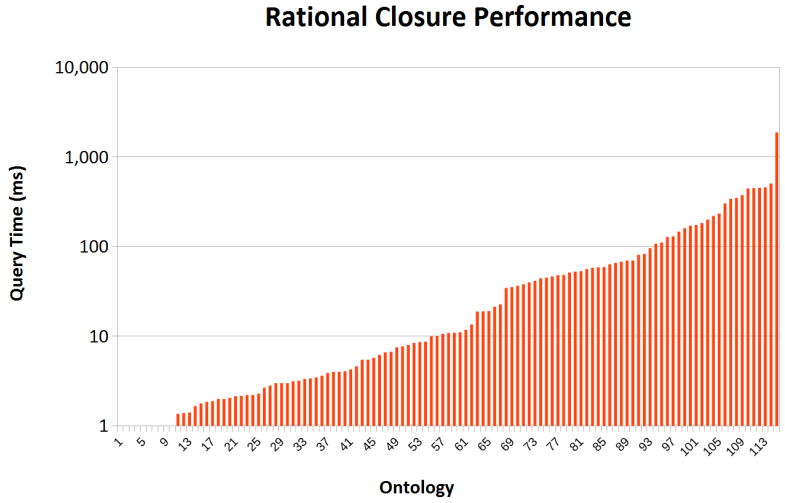
Rational and Lexicographic Closure

For Rational Closure, all queries (except those for Ontology 134) in the modified real world data could be executed in less than a second. On average over all ontologies the query time was around 80ms and 90% of all queries could be executed in 200ms or less. For Lexicographic Closure the average query time was much slower (around 18 seconds) but this is because of isolated queries that were much slower than average. This is corroborated by the median query time of around 250ms and the 75th percentile of 1.9 seconds (75% of the queries could be executed in 1.9 seconds or less). The plot of the average query times for both Rational and Lexicographic Closure are presented in Figure 6.24.

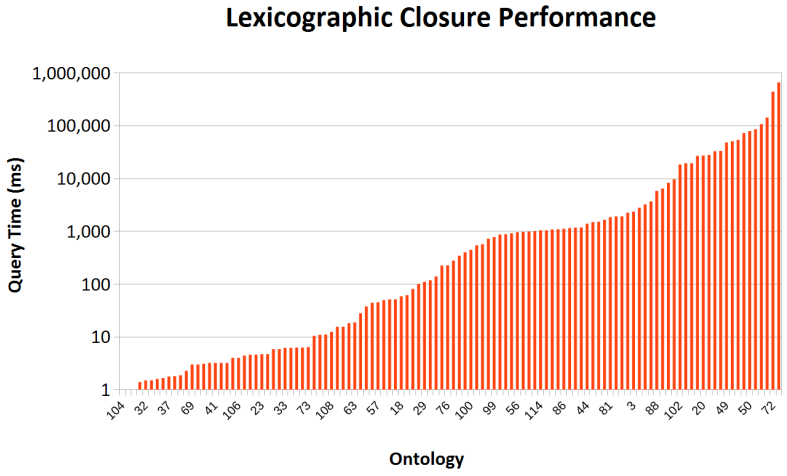
The data therefore confirms our analysis and generalisations in Section 6.1: that the performance of Rational Closure (even using our preliminary implementation) is feasible for TBox reasoning in modern ontology editing tools. The vast majority of queries terminate within 100ms. There are, however, a significant number of queries which take between 100 and 500ms to compute. This is in slight contrast to the results for the artificial data which found that less than 1% of all queries posed to the reasoner took longer than 100ms to compute (extremely few queries even approached close to 100ms).

We hypothesise that the main reason for the queries between 100 and 500ms is the much larger ontology sizes obtained in our dataset. In fact, we postulate that the sheer magnitude of ontology sizes in the data has the largest impact on the performance of Rational Closure. Figure 6.25 lends credence to this claim (both axes are of logarithmic scale).

However, even though the performance of Rational Closure decreases con-



(a) Rational Closure performance in the modified real world data.



(b) Lexicographic Closure performance in the modified real world data.

Figure 6.24: Average performance of Rational and Lexicographic Closure in the modified real world data.

siderably with the larger ontology sizes (the reason for this is that the performance of classical entailment also decreases as ontologies become larger), the performance still remains acceptable for practical reasoning tasks in ontology editing tools.

For the Lexicographic Closure we find that the performance on modified

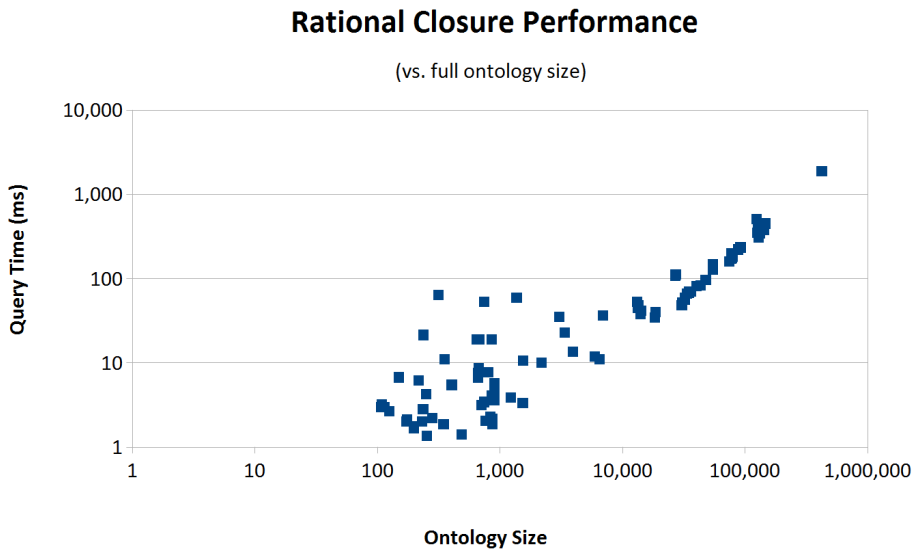


Figure 6.25: Average Rational Closure performance vs. ontology size in the modified real world dataset.

real world data is very similar to our artificial data (even considering the much larger ontology sizes in the former dataset). This is quite interesting because, considering that the major performance factor for Lexicographic Closure is problematic rank size, it must be the case that the problematic rank sizes encountered in the current dataset are comparable to those obtained in the artificial dataset. We plot the performance of Lexicographic Closure against problematic rank size in Figure 6.26 to illustrate this (the reader may compare the graph with Figure 6.10).

To illustrate that the problematic rank sizes of both datasets in our evaluation were very similar we give some figures: the mean values for the artificial and modified real world data were roughly 39 and 36 respectively, the median values 24 and 16, the 75th percentile values 58 and 39, the 90th percentile values 102 and 109 and the maximum values 236 and 320. There was, however, one anomalous ontology (Ontology 134) in the latter dataset which, for some queries, had a mammoth problematic rank of roughly 5900 axioms. However, we excluded the results for these queries from consideration because

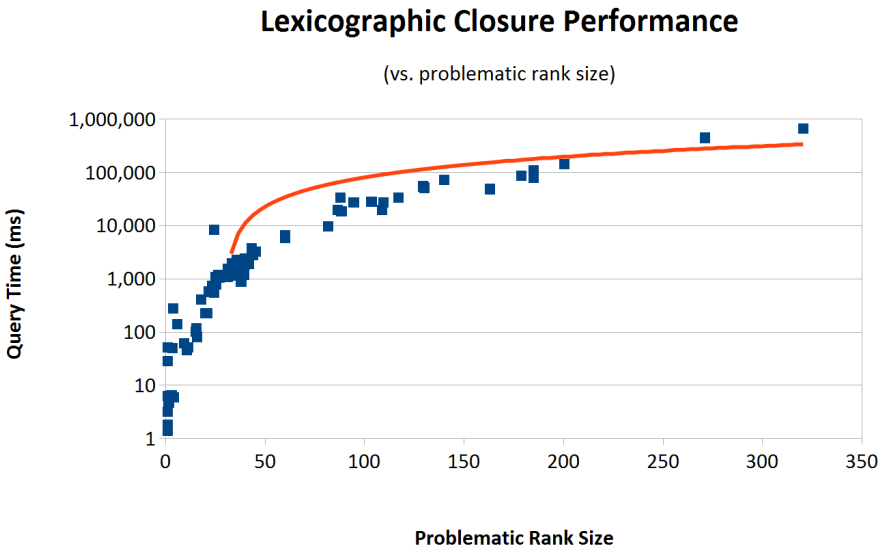


Figure 6.26: The performance of Lexicographic Closure is predominantly determined by problematic rank size for the modified real world data as well.

they require over $1,000$ seconds to compute.

Just like in Section 6.1 we would like to ascertain, for the current dataset, how much more expensive the Rational and Lexicographic Closures are than classical entailment. We find that, on average, the number of classical entailment checks required to check a defeasible entailment (using Rational Closure) is 2.7 and Lexicographic Closure is not much higher at 3.3. Since the most intensive component, by far, of the Rational Closure algorithm is its classical entailment checks, we can infer that Rational Closure takes roughly 2.7 times as long as classical entailment over the data.

For the Lexicographic Closure, just as in Section 6.1, the number of classical entailment checks is not much higher than Rational Closure. However, the algorithm requires computation of the LAC (which is an exponential problem in the size of the problematic rank). This intensive component of the algorithm means that we cannot base our comparison of Lexicographic Closure and classical entailment performance purely on the number of classical entailment checks. Therefore, we first compare the performance of Lexico-

graphic Closure to that of Rational Closure and then extrapolate an indirect comparison with the performance of classical entailment.

The 10th percentile for Lexicographic Closure is 2.5ms while the same percentile for Rational Closure is 1.39ms (Lexicographic Closure takes roughly twice as long). The 75th percentile value is 3 orders of magnitude higher than that of Rational Closure. From this we generalise that, for the majority of queries, the times for Lexicographic Closure are expected to be between twice as long as Rational Closure, to 3 orders of magnitude higher than those for Rational Closure. Rational Closure itself takes 2.7 times as long as classical entailment. Therefore, one can make the very general projection that Lexicographic Closure would roughly take between 5.4 and 2700 times as long as classical entailment (for the majority of queries).

Basic, Minimal and Lexicographically Relevant Closure

The results for the Relevant Closures follow a similar pattern to that of the artificial data. All three closures have almost identical performance over the data. Again, this is because justification computation forms the main component of all three algorithms and it consumes 99% of the reasoning time for each query. The overall performance is depicted in Figure 6.27.
Interestingly, whereas Lexicographically Relevant Closure was ever so slightly faster in the artificial data than the other two Relevant Closures, here it is slightly slower by a similar margin. We conjecture that the reason for this is that the problematic ranks are slightly larger in this dataset. In other words, after these problematic ranks are pruned (by removing the axioms from them that appear in the minimal C-bases) their sizes are still slightly larger than those in the artificial dataset.

We again imposed a timeout of $1,000$ seconds for each query and found that roughly 2% of them did not terminate within the time limit. For all cases it is no surprise that the major bottleneck for performance remains the computation of the justifications. Specifically, the bottleneck for computation of justifications is construction of the hitting set tree. We plot the query times

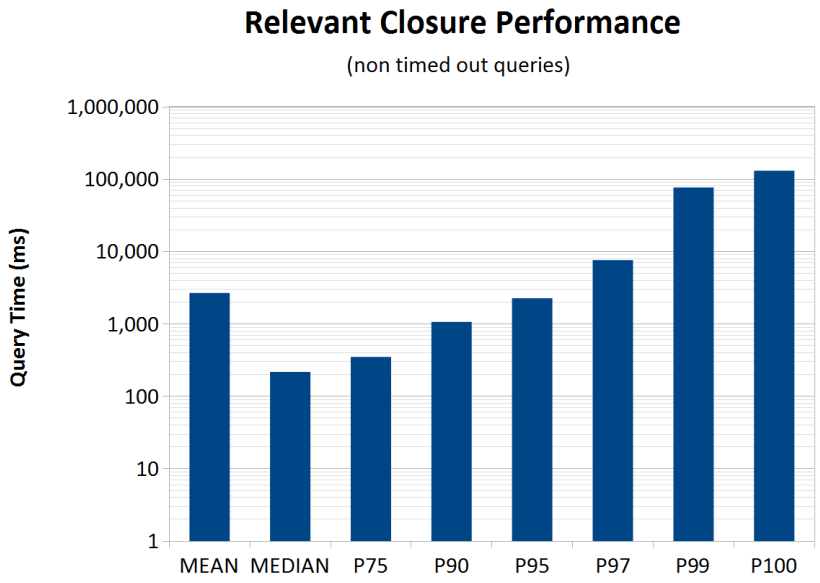


Figure 6.27: The performance of Basic Relevant Closure on the modified real world data. The other Relevant Closures have almost identical performance on the same data.

against the HST size in Figure 6.28.

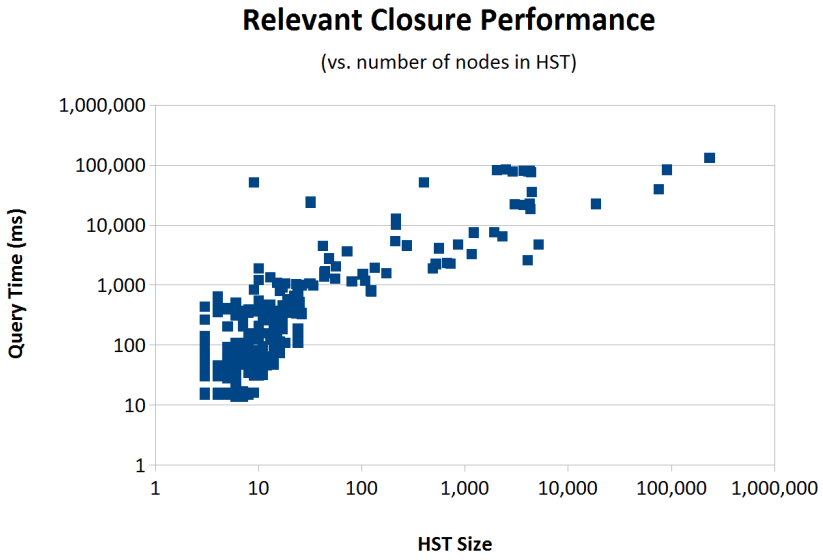
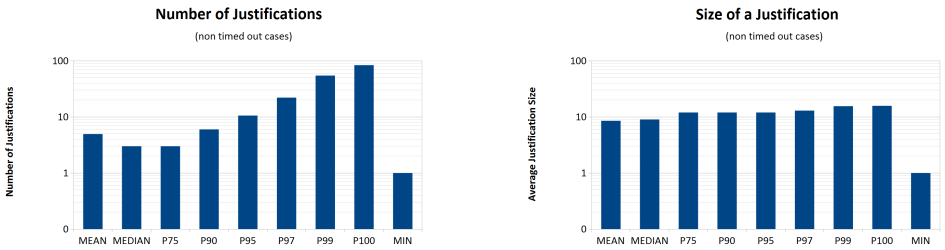


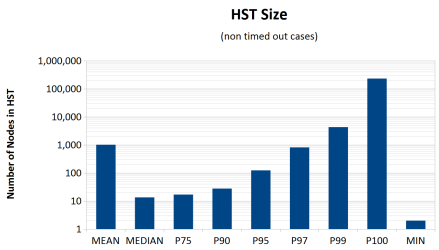
Figure 6.28: The major influence on Relevant Closure performance, for the modified real world data, is still the number of nodes in the hitting set tree.

Figure 6.28 clearly demonstrates that HST size has a major impact on the performance of the Relevant Closures. However, there are two outlier cases when the HST size is at 9 and 32. The corresponding query times for these cases are $51,405$ms and $24,881$ms respectively. The reason for these departures from the general trend is ontology size - these queries are executed on Ontology 134 in the dataset which has in excess of $400,000$ axioms (6010 of which are defeasible). This ontology is far larger than the vast majority of the other ontologies in the dataset.

To give the reader a sense of the justificatory structure of the non timed out cases, we plot the percentile occurrences of: number of justifications, size of a single justification and HST size for these queries in Figure 6.29.



(a) Number of justifications per query. (b) Average justification size per query.



(c) HST size per query.

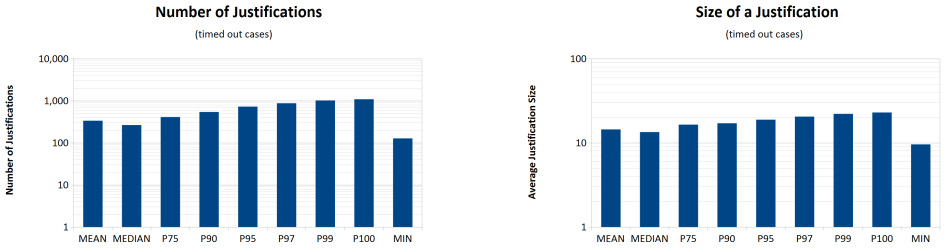
Figure 6.29: Justification metrics for the non timed out queries posed to the modified real world data.

Figure 6.29 gives some important insight into the complexity of justification computation that our algorithms are able to execute while still retaining reasonable defeasible inference times. We believe that any defeasible infer-

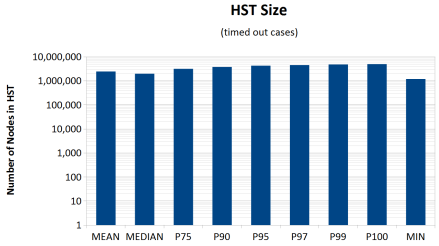
ence time of below 10 seconds (just higher than the 97th percentile value) is acceptable for our preliminary implementation of the Relevant Closures. Given this upper limit we can extrapolate that our algorithms would be able to compute just over 20 justifications with maximum HST sizes around $1,000$ to stay within this reasoning time.

In other words, for more complex justification computation (computing much more than 20 justifications and HST sizes closer to $10,000$ than to $1,000$) we cannot guarantee inference times lower than 10 seconds with the current implementation for the Relevant Closures.

We conclude our analysis of the performance of the Relevant Closures with an illustration of the justification metrics for the timed out queries in the modified real world dataset (see Figure 6.30).



(a) Number of justifications per query. (b) Average justification size per query.



(c) HST size per query.

Figure 6.30: Justification metrics for the timed out queries posed to the modified real world data.

The main discrepancies between the timed out cases and the non timed out cases are the number of justifications and the HST size. On average

the queries computed in excess of 300 justifications by the cut off time of 1,000 seconds. In fact, we have one case in excess of 1,000 justifications. In addition, the HST sizes grow to the millions of nodes by the cut off time. Interestingly, the average sizes of individual justifications stay very similar to the non timed out cases (around a maximum of 20).

In terms of relating the performance of Lexicographically Relevant Closure with that of its equivalent construction the Lexicographic Closure, Figures 6.27 and 6.24 reveal that they have very similar performance overall in the data.

In the artificial data, we witnessed that Lexicographic Closure was better in performance for 10% to 40% defeasibility. For the remainder of cases, the Relevant Closures (including Lexicographically Relevant Closure) were considerably better in performance.

Our explanation as to why the Lexicographically Relevant and Lexicographic Closures have more similar performances in this dataset, is that the justificatory complexity of the ontologies in this dataset is higher than that of the artificial data (we compute more justifications with larger HST sizes). Nevertheless, the problematic rank sizes encountered here are still similar to those in the artificial dataset. Therefore, the performance of Lexicographically Relevant Closure does not surpass that of Lexicographic Closure.

6.2.6 Discussion

We summarise the main observations and insights of our evaluation in the following subsections:

Rational and Lexicographic Closure

Rational Closure is the best performing of our presented defeasible reasoning algorithms and is suitable, even in its current implementation, for use in modern ontology editing tools. The data suggests that Rational Closure entailment testing takes roughly 3 times as long as classical entailment testing.

Specifically, the figures show that the mean times to test a defeasible entailment, using Rational Closure, are roughly between 25ms and 1880ms across both datasets. Therefore performance is well suited to reasoning in typical ontology development environments even without sophisticated optimisation.

As one would expect with the much larger ontology sizes in the real world dataset, the mean performance of Rational Closure increases from roughly 25ms in the artificial dataset to roughly 81ms in the real world data. Again, even though there is a considerable increase in query times in terms of magnitude, even the slowest times in the real world data are still fast enough to be considered suitable for modern ontology editing systems.

The major factor for Rational Closure performance has been shown to be ontology size (as inherited from classical entailment). The only other factor which could significantly affect Rational Closure is the length of the exception to exception chain. This has been shown to be fairly short, relative to the number of exceptions in the ontologies, in the artificial data (around a maximum of 16) and much shorter in the real world data (maximum of 3).

Although we do not know how long this chain would be in practice when real world defeasible ontologies become widely engineered, there is no evidence to suggest that the lengths would be wildly different to the ranges we encountered in our datasets.

For the Lexicographic Closure we found that the data showed very large variance in query times between the median and worst cases (3 to 4 orders of magnitude). Therefore, it is harder to evaluate at this juncture how feasible its performance would be for modern ontology editing systems.

The major performance factor was shown to be the size of the problematic rank and we could extrapolate that, to keep its performance very practical (say below 10 seconds per query), the problematic rank size should be kept lower than roughly 100 axioms. A very effective way to curb the problematic rank size is to not introduce too many defeasible subsumptions into the ontology. In other words, to keep the defeasibility of the ontology very low. This remains a major open question: what would the percentage defeasibility

values be for real world defeasible ontologies to be engineered in the future?

If these values would be very low (say lower than 10%), and the overall sizes of the defeasible ontologies are similar to existing classical ontologies, then we project that Lexicographic Closure performance would be feasible for the vast majority of data.

Even though we found Lexicographic Closure to be the worst performing of our presented algorithms (as we anticipated), we saw that its overall performance was hampered by isolated cases which were extremely hard (large ontologies with high problematic rank sizes).

If we have to compare its performance with that of Rational Closure (the highest average query times obtained with Rational Closure were around 1.9 seconds), we actually find that the 55th percentile value for Lexicographic Closure is 1878ms (very close to 1.9 seconds). In other words, 55% of the queries posed to Lexicographic Closure could be processed in times very similar to Rational Closure overall.

Its performance was also quite consistent across both datasets: we found that the mean times obtained in the real world data were roughly 5% faster than in the artificial data.

The Relevant Closures

The performance of the Relevant Closures is almost exclusively determined by the performance of justification computation. Since the performance of justification computation is highly sensitive to justificatory structure, and the latter is relatively unpredictable in real world ontologies, the performance of the Relevant Closures should be similarly unpredictable.

The data also gives some insights as to what kinds of justificatory structure are complex for reasoning. We found that the performance of the Relevant Closures is acceptable (below 10 seconds per query) if the maximum number of justifications that we need to compute does not increase significantly above 20 and the accompanying HST sizes do not grow considerably above the 1000 mark.

In fact, we found that the reasoning times reduce by around 80% when we concentrate only on queries which have these justificatory structures. A positive observation is that the majority of queries fell into this particular justificatory structure bracket. In other words, the harder cases where we had to compute more than 20 justifications with HST sizes much higher than 1000 occurred much less frequently in the data.

Interestingly, the mean times for the Relevant Closures were roughly 24% faster in the real world data than in the artificial data.

Our results suggest that, *overall*, the Lexicographically Relevant Closure is not a very effective optimisation for Lexicographic Closure. However, the data shows that when the justificatory structure of the ontology is simple (entailments have less than 20 justifications with HST sizes not much higher than 1000) then the Lexicographically Relevant Closure mean times are reduced by 82% in the real world data and by 81% in the artificial data.

In the end it is difficult to evaluate the suitability of the Relevant Closures, in terms of performance, for current ontology editing systems. The main reason is the unpredictability of justificatory structure which is the major influence in performance here.

Concluding remarks

In conclusion, we find that Rational Closure is the hero in terms of performance and the villains are the Lexicographic Closure and Relevant Closures. The main question that needs to be addressed, before the latter algorithms can be exonerated from (or bound to) their unfortunate title, is percentage defeasibility. If we know how much defeasibility is going to be used in real world ontologies, we would be in a better position to predict the performance of these algorithms.

While Rational Closure is perfectly suitable even its current implementation for use in current ontology editing tools, one would like to ascertain if its inferential caution is a drawback *in practice*. In other words there is scope for another investigative thread in this area to study the trade off be-

tween the inferential power of a defeasible entailment regime and its practical performance.

Finally, regardless of percentage defeasibility and other factors influencing reasoning performance, there is still a lot of scope for optimisation for all the algorithms that we have presented in this thesis. We hope that this evaluation sheds some light on which areas we need to concentrate on for this purpose.

Chapter 7

DIP: Defeasible-Inference Platform

In the previous chapter we showed that we do not pay an inordinate price for the extra expressivity of defeasible subsumption and basic defeasible reasoning tasks (based on the preferential reasoning foundation). Even though our algorithms still leave considerable room for optimisation, the performance of Rational Closure in particular is perfectly acceptable to be used in current ontology engineering environments, even in its existing state.

The performances of the Lexicographic and Relevant Closures are not as promising as Rational Closure but they still appear sufficient to be used, at least in an experimental manner, on small to medium-sized ontologies.

These algorithms can only benefit (in terms of optimisation) from implementation and integration into widely used ontology editing tools, and with a view towards faster implementations we have developed a plugin, *Defeasible-Inference Platform* (DIP), for the ontology editor Protégé including the aforementioned preferential reasoning algorithms.

In this chapter, we give an overview of DIP including its functionality, architecture and user interface (UI) design. We first give a brief overview of Protégé and the OWL API (the API Protégé uses to manipulate OWL ontologies), after which we introduce DIP and its integration into Protégé.

Finally, we give helpful instructions on how to access and use DIP in Protégé.

7.1 Protégé

Protégé is a free and open source ontology editor and knowledge base framework, originally developed by the Stanford Center for Biomedical Informatics Research at the Stanford University School of Medicine.

There are now four major versions, Protégé 3.x, Protégé 4.x, Protégé Desktop[1] (or Protégé 5) and WebProtégé[2]. Protégé 3.x is a legacy version of Protégé providing support for working with older frame-based [114] ontologies along with ontologies expressed using the now dated OWL 1 standard.

Protégé 4 was developed as part of the "CO-ODE" project by the University of Manchester in collaboration with Stanford Medical Informatics between 2007 and 2009. The Protégé 4.x versions do not allow frame-based ontology editing and were tailored for the shift to the next version of OWL, OWL 2 (which became a W3C recommendation in December, 2012).

Finally, incremental improvements to Protégé 4 between 2009 and 2013 eventually led to a new major release in 2014 - Protégé Desktop. When we use the term Protégé, in the remainder of this chapter, we are referring to the latest versions namely Protégé 4 and Protégé 5.

7.1.1 User Interface

Protégé makes use of a "tabbed" graphical user interface (see Figure 7.1). Each tab displays a different "ontology view". The most frequently used tabs are the following:

- Classes
- Object Properties

[1] protege.stanford.edu

[2] webprotege.stanford.edu

- Individuals
- Entities

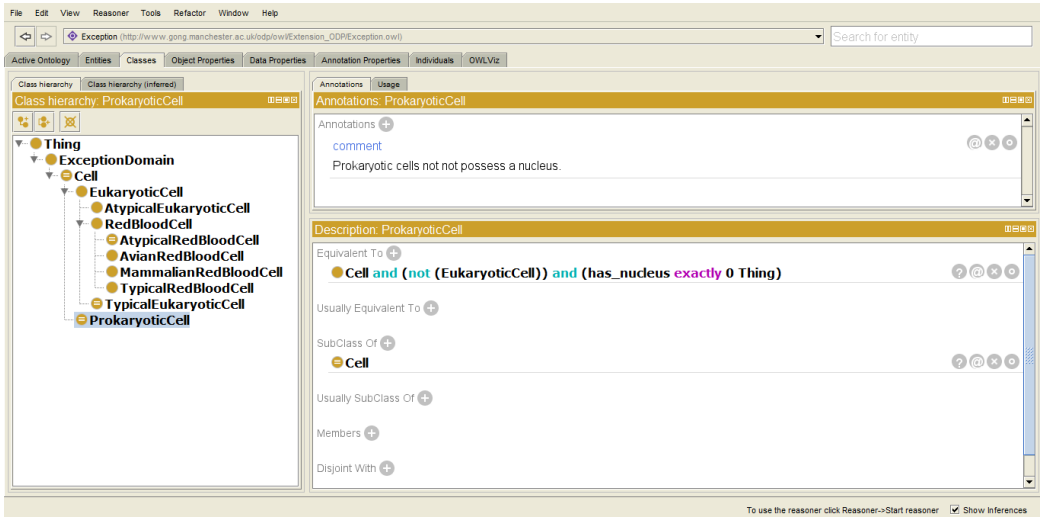


Figure 7.1: Protégé 4 ontology editor.

The Classes tab offers a class or concept driven view of the ontology. Here, the focus is on viewing the class hierarchy of the ontology, that is, the subsumption relationship between class names added to the ontology by the ontology engineer.

One can see an example of this hierarchy on the left hand side of Figure 7.1. The class names are arranged in a "'tabbed" format to indicate their subsumption relationship with other class names. That is, class names that are aligned vertically are sibling classes. Those that are indented with respect to others are subclasses of these others.

The "asserted" class hierarchy tab displays the taxonomy of the class names in the ontology *as specified by the ontology engineer*, as opposed to the "inferred" class hierarchy which is the (possibly) revised hierarchy after inferences made by a reasoner are incorporated into the asserted hierarchy.

Class names in Protégé are indicated by a preceding tan-coloured circular icon in the interface. The Object Properties tab displays the role names

(object property is OWL lingo for role name) in the ontology and the relevant information pertaining to them. Object properties are differentiated from other entities in the ontology by a preceding blue-coloured rectangular icon.

The Individuals tab displays the ABox information about the instances or individuals in the ontology. Individuals are indicated by a preceding purple-coloured, diamond-shaped icon.

The Entities tab combines the previously discussed tabs so that all the perspectives can be viewed and browsed simultaneously.

Protégé has support for a variety of ontology reasoners which can be installed as plugins in the Protégé system and be accessed via the "Reasoner" menu in the main toolbar.

To indicate inferences drawn from the ontology by installed reasoners, Protégé makes effective use of color in the interface. For example, after the reasoner classifies the ontology, the inferred subclasses, superclasses, equivalent classes, ABox assertions are displayed against a pale yellow background in each of the relevant tabs. Also, concept names which are found to be unsatisfiable in the ontology are displayed using red text.

Since Protégé allows working with multiple ontologies simultaneously, there is an "Active Ontology" tab displaying details and metadata about the ontology currently being worked on. There are numerous manuals and helpful resources on the Web for getting started with using Protégé[3].

7.1.2 The OWL API

Since Protégé is built primarily for OWL ontologies, an API is needed for loading, parsing and manipulating ontologies expressed in OWL.

The OWL API [95] is a Java-based API developed for creating, editing and managing OWL ontologies. Version 4 (the latest as of writing) follows, very closely, the OWL 2 structural specification recommended by the W3C.

Protégé itself has a very modular structure. In fact, it can be seen as set of plugins that interact together to provide the functionality of the complete

[3] `protege.stanford.edu/support.php#documentationSupport`

system. It includes, as one of its modules, the OWL API as its API of choice for loading and manipulating OWL ontologies (though there are alternatives available such as Jena[4] and Sofa[5]).

The reader is reminded that DLs form the logical underpinning of OWL languages and thus we can make some correlations between the constructs in these languages. Figure 7.1 illustrates a few of these correlations.

OWL Syntax	**DL Syntax**
owl:subClassOf	$\sqsubseteq$
owl:complementOf	$\neg$
owl:intersectionOf	$\sqcap$
owl:unionOf	$\sqcup$
owl:someValuesFrom	$\exists$
owl:allValuesFrom	$\forall$

Table 7.1: Correlation between OWL and DL syntax.

In the OWL API, ontologies are viewed as sets of axioms and annotations (meta-information about entities in the ontology). The API serves as a set of interfaces for manipulating and reasoning with OWL ontologies.

The OWLOntologyManager interface in particular, provide the means for loading, editing and saving ontologies. The OWLOntologyManager also provides access to the OWLReasoner interface which, in turn, provides access to inference services of a particular reasoning implementation (e.g. FaCT++).

To construct the building blocks (individuals, concept names and role names) of an ontology, one can use the OWLDataFactory class, which can also be accessed through the OWLOntologyManager interface.

Finally, the OWLOntology interface provides access to all the axioms and entities in an ontology. It also allows for selective access to axioms by various criteria such as axiom type (equivalence axioms, subclass axioms, disjointness axioms etc.) or axiom signature.

[4] `jena.sourceforge.net`

[5] `sofa.projects.semwebcentral.org`

7.2 Defeasible-Inference Platform (DIP)

DIP[6] is a plugin for Protégé providing the capability of representing defeasible subsumption statements in OWL ontologies, as well as reasoning with the resulting defeasible ontologies using the algorithms presented in this thesis.

We discuss the aspects of DIP allowing us to *represent* defeasible subsumption in OWL first, and thereafter we expand on the core functionality - the defeasible reasoning engine.

7.2.1 Expressing Defeasible Subsumption

Of course, as we have stated numerous times in Chapter 6, OWL does not yet natively support defeasible subsumption. To get around this issue, we employ the use of *OWL annotations* which are metadata, non-logical constructs in OWL, to indicate whether a selected classical subsumption in Protégé should be treated as defeasible by DIP.

For example, in order to represent the defeasible subsumption $\mathsf{Student} \mathrel{\sim\!\!\sqsubseteq} \neg(\exists \mathsf{receives.TaxInvoice})$, a modeller would add the classical subsumption $\mathsf{Student} \sqsubseteq \neg(\exists \mathsf{receives.TaxInvoice})$ to the ontology in Protégé, and then "toggle" this subsumption to be defeasible by pressing the button labelled "d" on the widget housing this subsumption in Protégé (see Figure 7.2).

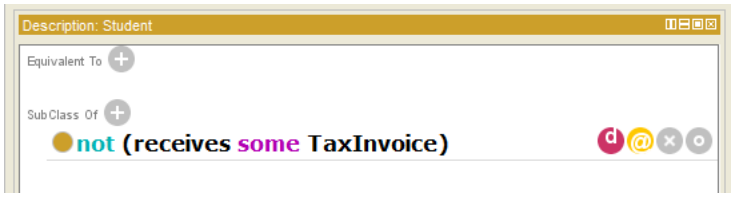


Figure 7.2: The class description pane for the class name Student in Protégé. The button labelled "d" in the figure is the extra feature added by DIP which allows one to toggle the attached axiom as defeasible.

The mechanism by which this is achieved, is that an annotation is added to (or removed from) the axiom which indicates that it is meant to be in-

[6] `github.com/kodymoodley/defeasibleinferenceplatform`

terpreted as defeasible. If this annotation is present then DIP considers the subsumption in its defeasible reading, if the annotation is not present then the axiom is considered classical by DIP. The following snippet of text shows how the defeasible annotation is attached to the intended axiom in OWL:

```
<SubClassOf>
        <Annotation>
            <AnnotationProperty abbreviatedIRI="defeasible"/>
            <Literal datatypeIRI="&xsd;boolean">true</Literal>
        </Annotation>
        <Class abbreviatedIRI="Student"/>
        <ObjectComplementOf>
            <ObjectSomeValuesFrom>
                <ObjectProperty abbreviatedIRI="receives"/>
                <Class abbreviatedIRI="TaxInvoice"/>
            </ObjectSomeValuesFrom>
        </ObjectComplementOf>
</SubClassOf>
```

In Figure 7.3 we see the graphical rendering of the above annotation in Protégé. We point out that DIP extends Protégé's axiom rendering capability to display the "UsuallySubClassOf" keyword (as shown in the figure) but this is purely a rendering tweak and, from the perspective of OWL and any classical reasoning implementation, the axiom is still considered classical.

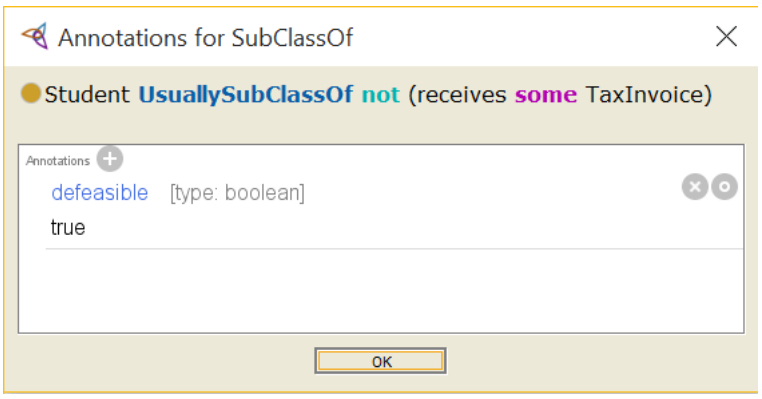


Figure 7.3: Graphical rendering of a defeasible annotation property in DIP.

Now that we have demonstrated how to represent defeasible subsumption in Protégé using DIP features, we can move on to discussing the core reasoning capabilities and features of DIP.

7.2.2 Reasoning Facilities

The most popular inferential insight into classical ontologies is usually their concept hierarchy (see Section 2.1). The reasoning service which computes this artefact is called classification because it reveals the implicit subset and superset relationships between all concept names in the ontology.

Classification typically requires many classical entailment checks. In fact, the focus in optimisations for classification is how to reduce this number of entailment checks from the worst case. Recalling that we need multiple classical entailment checks to perform a single defeasible entailment check, it becomes clear that classification for ontologies with defeasible subsumption can become impractical with our relatively unoptimised defeasible reasoning algorithms.

Furthermore, we cannot inherit many optimisations for classification from the classical case because we do not have some desirable properties of classical subsumption, like transitivity for example: if $C \sqsubseteq D$ and $D \sqsubseteq E$ are in the Rational Closure of a KB, it does not necessarily mean that $C \sqsubseteq E$ is also in the Rational Closure of the KB.

We promote a different reasoning task as the core reasoning service of DIP - the querying for the defeasible subclass and (or) superclasses of a given class expression. That is, given a user specified class expression C constructed from the vocabulary of the ontology, DIP is used to compute all (class) names A in the ontology s.t. either $C \sqsubseteq A$ or $A \sqsubseteq C$ follows from the ontology (using a selected preferential reasoning algorithm).

This "interaction model" with the reasoner is actually used, in the classical case, by the popular Protégé plugin called the DL Query Tab[7]. The DL Query Tab, which was developed by Matthew Horridge at the University of Manchester, comes pre-installed in most current versions of Protégé.

Horridge's tool is able to compute the set of all named classes in the given ontology that are either subclasses, superclasses or equivalent classes with the user specified class expression. It is also able to give the ABox

[7] protegewiki.stanford.edu/wiki/DLQueryTab

instances of the given expression. For DIP we have chosen to focus purely on defeasible subclasses and superclasses (the natural analogues to classical subclasses and superclasses).

The reason for not considering ABoxes (as we mentioned earlier in this thesis) is that ABox algorithms, in the preferential case, are not yet mature.

We hold that this task for preferential reasoning algorithms is still helpful to gain insight into ontologies containing defeasible subsumption. For example, it is quite natural, given an ontology describing species of birds, to ask the reasoner which ones usually don't fly.

This can be accomplished using DIP's interaction model by supplying the class expression $\neg(\exists \mathsf{ability}.\mathsf{FlyingAbility})$ and then asking for all mentioned class names A s.t. $A \tilde{\sqsubseteq} \neg(\exists \mathsf{ability}.\mathsf{FlyingAbility})$ follows from the ontology.

DIP would then, for example, return class names such as Emu, Ostrich and Penguin if such terms are appropriately described in the ontology. This task returns the defeasible *subclasses* of the given expression.

In contrast, one might also want to ask for the defeasible *super*classes of a given expression. For example, one might ask for the typical attributes of students (Student) in an ontology describing university students. DIP should then return attributes such as NonTaxPayer, Poor and UnMarried (students are usually non-tax payers, poor and unmarried).

We re-emphasise that DIP will return only (class) *names* as results. This is mainly to ensure better performance because there are far fewer named classes in an ontology, than there are complex class expressions that can be constructed with the vocabulary.

Therefore, in ontologies which are appropriately designed for this purpose, a reasoning task that is increasing in popularity is the querying for required *fillers* (non-complex ones) of existentially quantified roles. Matthew Horridge augmented the DL Query Tab with this functionality to address the demand for such a reasoning service in Protégé[8].

Recently, this service was exploited to aid in the taxonomic revision of

[8]github.com/protegeproject/existentialquery

afrotropical bee species [143]. Here we devise an example setting of our own to illustrate the usefulness of this reasoning task:

Example 31 *Consider the following defeasible KB:*

$$\mathcal{T} = \left\{ \begin{array}{lcl} \mathsf{Dish} & \equiv & \exists \mathsf{hasIngredient}.\top \sqcap \exists \mathsf{hasTasteQuality}.\top, \\ \mathsf{SushiDish} & \sqsubseteq & \mathsf{Dish} \sqcap \exists \mathsf{hasFilling}.\top, \\ \mathsf{VegMakimono} \sqcup \mathsf{SweetSushi} & \sqsubseteq & \mathsf{SushiDish}, \\ \mathsf{VegMakimono} \sqcup \mathsf{SweetSushi} & \sqsubseteq & \neg(\exists \mathsf{hasIngredient}.\mathsf{Fish}), \\ \mathsf{hasWrapping} & \sqsubseteq & \mathsf{hasIngredient}, \\ \mathsf{hasFilling} & \sqsubseteq & \mathsf{hasIngredient}, \\ \mathsf{hasCondiment} & \sqsubseteq & \mathsf{hasIngredient} \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{lcl} \mathsf{SushiDish} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{hasWrapping}.\top \sqcap \exists \mathsf{hasCondiment}.\top, \\ \mathsf{SushiDish} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{hasIngredient}.\mathsf{Fish}, \\ \mathsf{SushiDish} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{hasIngredient}.\mathsf{SoySauce}, \\ \mathsf{SushiDish} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{hasIngredient}.\mathsf{Ginger}, \\ \mathsf{SushiDish} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{hasTasteQuality}.\mathsf{Salt}, \\ \mathsf{SweetSushi} & \mathrel{\tilde{\sqsubseteq}} & \forall \mathsf{hasFilling}.\mathsf{Fruit}, \\ \mathsf{SweetSushi} & \mathrel{\tilde{\sqsubseteq}} & \exists \mathsf{hasTasteQuality}.\mathsf{Sweet}, \\ \mathsf{SweetSushi} & \mathrel{\tilde{\sqsubseteq}} & \neg(\exists \mathsf{hasTasteQuality}.\mathsf{Salt}) \end{array} \right\}$$

□

Regarding the knowledge represented in Example 31 on Page 329, one can pose the natural language question "what are the typical properties of sushi?". In DL terms this question can be answered by querying the KB for the defeasible superclasses of SushiDish.

Rational Closure will endorse that, structurally, sushi is usually composed of an external wrapping and is usually also accompanied with condiments ($\exists \mathsf{hasWrapping}.\top \sqcap \exists \mathsf{hasCondiment}.\top$). It will also endorse that sushi usually has a salty taste even though there is an exception to this - sweet sushi.

It will also endorse that sushi usually has the ingredients fish, ginger and soy sauce even though there are two exceptional cases in the ontology - sweet sushi and vegetarian makimono, which don't contain fish.

However, since DIP (like the DL Query Tab) will only return named classes as results, we will not return any results for the expression SushiDish

because there are no named classes in the ontology that are defeasible superclasses of SushiDish.

Rather, the user is encouraged to ask more specific questions such as: "what are the typical *ingredients* of sushi?" or "what is the typical *taste* of sushi?". In essence the user is asked to supply a role name (for example hasIngredient or hasTasteQuality) in addition to the query class expression.

In our example, we can identify the typical ingredients and taste of sushi by asking DIP if the axioms $\mathsf{SushiDish} \mathrel{\sqsubseteq\!\!\!\sim} \exists \mathsf{hasIngredient}.\mathsf{X}$ and $\mathsf{SushiDish} \mathrel{\sqsubseteq\!\!\!\sim} \exists \mathsf{hasTasteQuality}.\mathsf{Y}$ follow from our ontology (according to the selected closure), for all named classes X and Y in the ontology.

The set of all X's and Y's returned by DIP will constitute the typical ingredients and taste of sushi according the preferential closure algorithm currently selected in DIP.

7.2.3 Interface

We have chosen to design the interface for DIP to closely resemble the DL Query Tab by Matthew Horridge, i.e., by bundling most of its reasoning functionality in a separate tab in the Protégé UI.

Classical DL reasoners are integrated into Protégé in a far less conspicuous manner. Because most of these reasoners implement the standard OWLReasoner interface from the OWL API, Protégé communicates their reasoning results to the user (in terms of the UI display) in the same way.

Since DIP is supposed to reason with the extra construct of defeasible subsumption, it is not possible to implement the OWLReasoner interface alone. One would have to also extend the OWL API to be able to distinguish between classical and defeasible subsumption statements.

We decided that this is not a feasible undertaking at present, but it is a worthwhile project for the future if defeasible reasoning becomes widely used in practical settings. Therefore, we adopt the approach of the DL Query Tab since the defeasible analogues of its reasoning tasks are potentially very useful to gain insight into defeasible ontologies.

In Figure 7.4 we give a screenshot of the DIP Tab in Protégé.

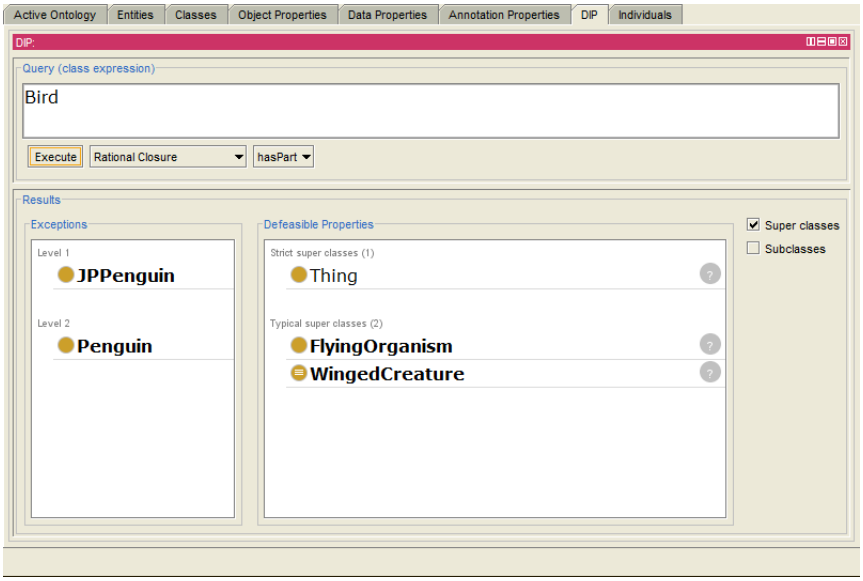


Figure 7.4: The control panel interface of the DIP tab in Protégé.

At the top of Figure 7.4 we see the text box labelled "Query (class expression)". Here the user is required to specify a class expression constructed from the vocabulary of the loaded ontology.

Below this text box there is a button labelled "Execute" for executing the reasoning task and displaying the results in the two panels below this button. On the right hand side of the execute button, there are two drop-down lists. The first is the reasoning algorithm menu from which the user would select the desired closure algorithm to use, i.e., either Rational, Lexicographic, or one of the Relevant Closures.

The second drop-down list contains the role names (object properties in OWL-speak) mentioned in the ontology. If desired, we can select a role name from this list to focus our reasoning task specifically on the *fillers* of this role which satisfy our provided class expression query (see Example 31 on Page 329).

The two results panels display the list of exceptions in the ontology and the results of our query, respectively. The list of exceptions corresponds

to the LHS concepts of the axioms in the exceptionality ranking that we discussed in Chapter 4.

The panel labelled "Defeasible Properties" partitions the results of our query into two sections: the *strict* subclasses (or superclasses) and the *defeasible* subclasses (or superclasses), of the provided class expression.

According to all the preferential reasoning algorithms presented in this thesis, the strict results are those that follow (using classical reasoning) from the TBox itself (excluding the DTBox).

On the far right hand side of Figure 7.4, we have two checkboxes for the user to indicate whether he or she would like to see either the subclasses or superclasses of the given query (or both).

For convenience, we have also provided two panels on the DIP tab to display the list of defeasible and strict axioms in the ontology (see Figure 7.5).

Figure 7.5: Panels provided in the DIP tab to display the list of defeasible and strict axioms present in the ontology.

The defeasible axioms are each rendered against a pink background, while the strict axioms are each rendered against a light green background.

7.2.4 Architecture

One of the most attractive features of DIP is that its algorithms reduce to classical reasoning procedures. Therefore, as long as there is a sound and

complete classical DL reasoner selected in Protégé, DIP will use this reasoner to perform the defeasible inferences it requires.
Here we give an informative diagram (Figure 7.6) which highlights DIP's interaction with a given classical reasoning implementation, as well as emphasising the architectural composition of the tool, and how the different components work together.

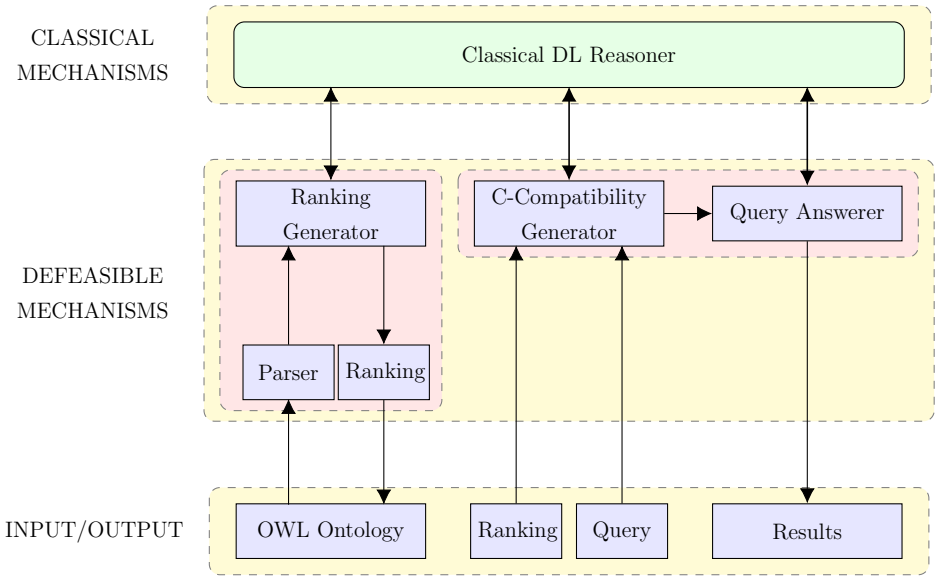


Figure 7.6: A high-level architectural view of DIP's components and their interaction.

As we have pointed out in Chapter 4, concept exceptionality is a central principle in preferential reasoning. Through concept exceptionality one can discern an a priori exceptionality ranking (also discussed Chapter 4) of the LHS concepts in the ontology. DIP thus has a "Ranking Generator" component which computes this ranking for every stable version of the ontology.

Whenever the ontology is modified (i.e., logical axioms are added, edited or removed), DIP recomputes this exceptionality ranking to reflect the changes. The loaded ontology first has to go through a simple "Parser" component to identify and separate the defeasible subsumption statements from the classical statements. Once the ranking is computed, it is stored to the ontology file. Again, this can be accomplished by using the ever versatile OWL an-

notation. Concretely speaking, DIP annotates each axiom in the ontology with the number representing its rank in the computed ranking (if the axiom actually appears in the ranking).

This feature is useful because when the ontology is reloaded from file at a later stage, if the ontology has not been modified, then we do not have to recompute the ranking. We need only load the ranking by analysing the annotations of the defeasible axioms in the ontology. This behaviour of serialising the ranking is represented by the arrow between the "Ranking" and "OWL Ontology" components.

Here we also make clear that the ranking of the ontology is independent of a query or reasoning task that needs to be executed. Therefore, the computation of the ranking is viewed as an "off-line" task. This is the reason why we do not have an arrow connecting "Ranking Generator" to "C-Compatibility Generator" in Figure 7.6.

When reasoning needs to take place, we require the ranking itself and a query (for example a class expression) as input for the "C-Compatibility Generator" component. The implementation of this component is actually the major differentiating factor between the preferential reasoning algorithms. Here the algorithms try to identify the maximal subset of the ranking which preserves that the query is *not* exceptional (or can be considered normal). As we know from Chapter 4, the algorithms base their notions of "maximal" on different principles.

For example, Rational Closure regards each rank in the ranking as the most elementary "blocks" of knowledge to play around with when it decides on the maximal knowledge to retain. Lexicographic Closure breaks these blocks into smaller chunks and decides on which permutations of them to retain. The Basic Relevant Closure only retains axioms that are known *not* to contribute to the exceptionality of the query (by using justifications). The Minimal Basic Relevant Closure allows the retaining of axioms that contribute to the exceptionality of the query, except those that are of lowest rank in the ranking (i.e., the least exceptional ones). Finally, the

Lexicographically-Relevant Closure combines the latter approach with Lexicographic Closure for possible optimisation gains.
Once the C-Compatible subset of the ranking is identified, this artefact is sent to a "Query Answerer" module which liaises with a classical reasoning implementation (the currently selected one in the context of Protégé) to answer the user-posed query. The "Query Answerer" component deals with actually formulating the appropriate question (reasoning task) to ask the classical reasoner based on the defeasible query posed by the user.

In the case of DIP we advocate the computation of the defeasible subclasses or superclasses of a given class and the latter task can, for example, be formulated as a classification problem in terms of classical reasoning. In other words, if we would like to identify the defeasible superclasses of a given class C, then in essence we are asking for all names X in the ontology such that $C \mathrel{\tilde{\sqsubseteq}} X$ follows from the ontology, according to the selected closure.

We saw in Chapter 4 that, once the C-Compatible subset of the ranking is obtained, then $C \mathrel{\tilde{\sqsubseteq}} X$ follows from the *original ontology* if and only if $C \sqsubseteq X$ follows classically from the classical translation of the C-Compatible subset. Therefore, after classifying the classical translation of the C-Compatible subset we can "read off" the superclasses of C as our results.

The defeasible *subclasses* are a bit more performance intensive to obtain because, in the worst case, we have to obtain X-Compatibility for *each* name X in the ontology (to determine if $X \mathrel{\tilde{\sqsubseteq}} C$ follows from the ontology).

The "Query Answerer" component is abstractly developed to cater for the possible future development of defeasible queries that are different to the subclass/superclass ones that we discuss above. In other words, the reasoning task that we describe above is just one possible task that is useful in the defeasible context. In future we may identify novel and interesting reasoning tasks that give other useful insights into defeasible ontologies.

7.3 Discussion

We have presented DIP as a prototype tool for reasoning over the terminological part of defeasible ontologies. We believe that it provides a core reasoning service which is beneficial to use, at least experimentally, for examining and comparing the inferential character of preferential reasoning algorithms. While DIP provides the core reasoning task of computing defeasible subclasses and superclasses of a given expression, there may prove to be other compelling reasoning tasks giving insight into defeasible ontologies.

There are also a host of other features which one can imagine would be complementary to DIP. For example, since defeasible subsumption is not transitive in general (contraposition also does not hold in general), the problem of developing optimised classification algorithms in this context is a non-trivial problem. Further to this, for the same reasons just mentioned, it is not clear how to best *visualise* and display the defeasible class hierarchy to an end user. Of course, a directed graph of some kind might suffice but the necessary formalisation thereof still needs to be agreed upon.

Chapter 8

Conclusions

We conclude this thesis with a summary of the main contributions, a discussion about some outstanding issues that could not be addressed as well as a presentation of suggestions for interesting future work.

8.1 Summary of Contributions

We set out with a pragmatic mindset to make the first steps in taking the state-of-the-art of defeasible reasoning theory (for DLs) into the practical sphere. We chose to focus on the preferential reasoning approach in this regard for a variety of reasons.

It has an elegance that speaks well to intuition without losing its strong logical character. It also defines quite an abstract framework for defeasible reasoning which gives it a certain robustness.

This generality in the formalism (and its outlook of studying the structure of rational consequence relations) makes it useful as a lens through which one can study other defeasible reasoning formalisms.

What we have essentially done in this thesis is motivate that the preferential reasoning approach is, in its current state, "ready" to be transferred to the practical setting.

We do not claim that all theoretical aspects have been resolved yet. In

fact we will show some of the main gaps still needing to be filled in this regard in Sections 8.2 and 8.3.

Our main thrust in this thesis is to show that preferential reasoning (for terminological knowledge) is practically viable in DLs from both a qualitative and quantitative perspective (despite the already known, and slightly disappointing, complexity results for some algorithms).

The main reason for promoting the application of preferential reasoning in practical settings is to stimulate feedback from users of DL-based systems on what areas need to be improved and investigated further, both from a theoretical and practical perspective.

Ultimately, this relationship between the users of defeasible reasoning formalisms, and the engineers of such formalisms, is going to be more intimate than with classical reasoning formalisms. This is because there is generally a much clearer consensus on what constitutes entailment in classical formalisms with a precise model-theoretic semantics.

Defeasible reasoning systems, on the other hand, are more contentious when it comes to entailment. Indeed, it seems likely that there is no single "best" defeasible entailment regime. In practice, different regimes may be suitable in different contexts.

With this spirit of breaching the divide between theory and practice for defeasible reasoning in DLs, we actually contribute concretely with the first known implementation of a system for preferential reasoning in DLs - Defeasible-Inference Platform (DIP).

It is clear from the composition and content of the thesis that we have taken a broad approach by addressing a variety of areas towards our ultimate aim (although not being comprehensive in terms of depth in any one area).

We list more specifically the individual contributions we have made below, together with their significances in the broader context of the thesis goals:

1. Starting out in Chapter 3 we have strengthened anecdotal arguments expressing that there is a need for defeasible features in DL-based languages used in practical settings. There we performed a preliminary

lexical analysis of ontology documents to reveal quantitative evidence supporting that, in general, there is a significant plea for defeasible representation in real world ontologies.

2. KLM argued convincingly, that even after introducing their notion of defeasible implication to propositional logic, that this relation should still retain some inferential properties of classical implication (when interpreted on the entailment level). However, their arguments have not been generalised to the DL case. In Chapter 3 we motivated the rationality of these postulates in the DL setting by taking into account the added structure of DLs (we had to take into account the concept semantics of DLs to supplement the arguments by KLM).

3. The central principle behind reasoning about exceptions in the preferential framework is the definition of concept exceptionality. The critical theoretical contribution which paved the way for practical implementation of preferential reasoning for DLs, is the reduction of concept exceptionality to classical DL entailment. To this end we have provided an alternative semantical characterisation for this reduction, using the notion of disjoint union of ranked models. This is done at the start of Chapter 4 and, even though the reduction is already expressed in the literature, our proofs showing this reduction are unique and more self-contained than others in the literature. A very strong advantage of our preferential algorithms, over other nonmonotonic formalisms, is that they all reduce fully to classical DL entailment allowing one to use off-the-shelf DL reasoners to perform defeasible inference.

4. In Chapter 4 we also provide a semantical characterisation for Lexicographic Closure for DLs, and a novel algorithm for computing it in this setting. We also go on to define an alternative way to compute Lexicographic Closure using the construction of *Lexicographically-Relevant* Closure which may be an optimisation for Lexicographic Closure in some practical settings.

5. Although it does not exactly fit the "practical reasoning" theme of this thesis, we have also contributed a defeasible notion of disjointness (as well as equivalence) of class expressions (see the end of Chapter 4). We have also motivated their intuition with examples and demonstrated that they can be represented using defeasible subsumption.

6. We also gave a few preliminary optimisations for computing preferential reasoning in DLs. In our experiments, the most impactful has been the demonstration of the relationship between concept incoherence and concept exceptionality (see Lemma 26 on Page 294). In other words we were able to show that, considering the classical counterpart of the ontology, only incoherent concepts have the potential to be exceptional (w.r.t. to the defeasible counterpart of the ontology). If a concept is not incoherent it can never be exceptional.

7. Chapter 5 gives the first evaluation of the major defeasible reasoning formalisms for DLs against the KLM postulates. This contribution makes the relationships between the inferential behaviours of the formalisms clearer. In particular, it highlights the attractiveness of the Lexicographic Closure as a defeasible reasoning algorithm striking a good balance between rational inferential behaviour *as well as* inferential strength.

8. In Chapter 6 we give a thorough performance evaluation of the reasoning algorithms presented in this thesis. The evaluation is only preliminary in the sense that our data is not wholly naturally occurring, and thus, one cannot be definitive in extrapolations from the data (although we can make reasonably strong generalisations from it). We found that the performance of Rational Closure is already well suited to practical use in real world ontology development settings. We found large discrepancies in performance between the average and worst cases for the Lexicographic and Relevant Closures. This was the main negative that we found in the evaluation. However, the mean times and

75th-percentile times for entailment checking using these closures were well below the one second mark. We believe this is very encouraging because (a) the algorithms use only two optimisations and (b) the majority of data (both synthetic and modified real world) are not considerably hard.

9. We have also identified the major bottlenecks and barriers to preferential reasoning performance in DLs. We found that all algorithms inherited the main (and obvious) performance influencer from classical reasoning - ontology size. In particular, we have uncovered a critical question which would reveal the fate of preferential reasoning performance if it were to be accepted in practical settings. This is the question of percentage defeasibility of ontologies. What fraction of the axioms in a real world ontology would contain defeasible statements? Answering this question would go far to reveal how much more work one needs to do (over and above classical reasoning) in order to perform defeasible inference. Rational Closure does not have a major performance bottle neck. We showed that its main performance influencer is the number of defeasible axioms (which is proportional to the number of potential exceptions in the ontology). The bottleneck for Lexicographic Closure is the computation of the LAC whose performance is highly dependent on problematic rank size. We found that, as this number increases close to, and beyond, the number 100, the performance drastically degrades to unacceptable levels. For the Relevant Closures we found that justification computation is by far the major performance influencer. For justification computation itself we found the major bottleneck to be the construction of the hitting set tree (the number of nodes in the tree is mainly determined by the degree of overlap in the justifications).

10. While sourcing data for our performance evaluation in Chapter 6, we provided a novel method for introducing defeasible features into OWL ontologies [49] thereby enriching the quality of the data (and hence

extrapolations from the data).

11. Our practical focus in this thesis culminated in the development of a plugin tool for the ontology editor Protégé for representing and reasoning with defeasible subsumption in modern OWL ontologies. The tool currently provides functionality for computing the defeasible subclasses and superclasses of a user-specified class expression.

8.2 Outstanding Issues

While we have contributed significantly to making preferential reasoning more practical for DLs, there are some areas that we could not address for pragmatic reasons. We list the most relevant issues not addressed here:

1. The most glaring omission of this thesis is the topic of preferential reasoning for the ABox. Although there are a number of proposals for the semantics of ABox preferential reasoning [50][71, Section 3.3], there are a number of issues which still need to be solidified both from a theoretical and practical perspective. In particular, the current algorithms require one to consider all possible orderings (sequences) of individuals explicitly mentioned in the ABox in order to construct preferential extensions of the ABox. The procedure has an EXPTIME-COMPLETE worst case complexity but it is clear that when a large number of individuals are mentioned in the ABox, the performance of the algorithm would drastically degrade.

2. While the theoretical foundation of our work assumes that we are working with the $\mathcal{ALC}$ concept language, our results are, in principle, applicable to a wide class of DLs ranging from $\mathcal{ALC}$ all the way up to $\mathcal{SHIQ}$ [74]. We have, however, not discussed in detail how our results are able to generalise to more expressive DLs than $\mathcal{ALC}$. Although there have been independent efforts to explore preferential reasoning in

low complexity DLs [73, 70, 53], we ourselves have not yet investigated this problem in detail.

3. The fact that we have not considered ABox reasoning in detail meant that we could not *comprehensively* compare alternative forms of defeasible reasoning for DLs. As we saw in Chapter 5, some major formalisms started off considering the problem of deriving defeasible ABox-like inferences from some defeasible background knowledge. In other words, considering some object of the domain, and some incomplete knowledge about this object, one can use some defeasible background knowledge about the domain to assign some additional plausible attributes to this object. For example, if I know of a student named *John* and I know that students usually don't pay taxes (and if I know nothing else), then one can plausibly derive that *John* does not pay taxes. In contrast, preferential reasoning took a different initial outlook: given some defeasible background knowledge about some domain of a terminological nature, what other defeasible knowledge (also of a terminological nature) can I derive from this knowledge? For example, if I know that students usually don't pay taxes, and I know that employed students are types of student, what should I conclude about the tax paying status of employed students in general?

4. Daniel Lehmann presented some informal properties of defeasible inference that he felt should be satisfied in general [116]. We ourselves have also motivated the sensibility of these properties in our particular generalisation of them to the DL case (see Chapter 3). However, Lehmann mentioned that these properties appear to have no purely formal representation. Because of the inability to formalise these properties, we have not been able to evaluate the formalisms mentioned in this thesis against them. However, if it is indeed the case that these properties have no formal representation, then we question the usefulness of stating and discussing them in the context of automated reasoning. In

other words, if we cannot make design decisions in the development of a defeasible formalism which can guarantee that these properties are satisfied, then what use are they? The only obvious use appears to be that, provided we agree that they are intuitive properties, we can show by counter-examples when a particular formalism does *not* satisfy them. However, we are more interested in formally demonstrating when a particular formalism *would* satisfy these conditions as well as designing formalisms from the ground up to respect them. Therefore, we believe there must exist either a formal, or at least semi-formal, way to represent these properties, and this still remains to be investigated.

5. On the subject of performance, we have left clues in Chapter 6 as to where possible optimisations for preferential reasoning might lie. We ourselves have used just two optimisations in this thesis: (a) ignorance of knowledge that is irrelevant to the query being posed (using modularisation), and (b) when identifying exceptions, ignoring knowledge that does not have the potential to contain exceptions (Lemma 26 on Page 294). However, despite these, our algorithms remain very modestly optimised compared with classical reasoning implementations. With regards to the compilation of the exceptionality ranking of an ontology, there is also tremendous scope for techniques to avoid naïve recomputation of the ranking every time the ontology is modified.

6. With regards to evaluating the performance of preferential reasoning, in Chapter 6 we devised a method for introducing defeasibility into real world ontologies. The method worked by identifying the potentially exceptional axioms in the ontology by identifying its classically incoherent concepts. We tried to "cater" for all these exceptions by converting all axioms related to the signature of the incoherent concepts, to defeasible axioms. This was not the most laconic way to introduce defeasibility because we may unnecessarily require that an axiom be defeasible. An interesting problem remains to be solved of whether we can introduce

a "minimal" number of defeasible axioms to the ontology in order to cater for all potential exceptions.

8.3 Future Work

Section 8.2 showed the areas that need to be addressed towards the goals of this thesis (i.e., within the scope of the broad "vision" of this thesis). Apart from earmarking these issues for investigation in future work, we also identify areas outside of the scope of this thesis (although related in topic) that would be interesting to address.

Defeasible Quantifiers: In this thesis we focused purely on one notion of defeasibility introduced into DLs - defeasible subsumption. In principle, one could also introduce notions of defeasibility for other constructs in DLs. Recently there has been a proposal by Britz et al. [43] for defeasible versions of the existential ($\exists$) and universal ($\forall$) role quantifiers in DLs.

Borrowing from the authors work, we have the example concept expression $\mathsf{Lawyer} \sqcap \forall \mathsf{hasClient}.\mathsf{PayingClient}$ which describes the set of all entities in our domain representing lawyers, all of whose clients are paying clients.

However, if we would like to refer to those lawyers who *normally* defend only paying clients, the above representation is too strong. We need an alternative representation capturing this latter intuition using defeasible universal role quantifiers ($\forall$).

Similarly, the representation $\mathsf{Lawyer} \sqcap \exists \mathsf{hasClient}.\mathsf{PayingClient}$ refers to those lawyers *each of which* has at least one paying client. However, with classical existential role quantifiers it is difficult to be able to capture: all those lawyers whose *normal* clientelé includes at least one paying client.

We wish to explore the integration of ($\forall$) and ($\exists$) into DLs with defeasible subsumption and the associated effects on reasoning.

Additional Inferential Postulates: In Chapter 3 we argued that the rationality postulates of KLM should be the *minimal* requirements of a rational defeasible entailment regime. However, we also remarked that there may be

other rules that may be suitable in various defeasible reasoning contexts.

One particular effort in this area is by Bezzazi et al. [26] who have explored, amongst other properties, variants of Rational Monotonicity and weakened versions of transitivity and contraposition, together with the inferential relationships between these properties.

Performance Benchmarking and Comparison: It would be highly beneficial for defeasible reasoning research if there could be a way to standardise a benchmark of ontology data that could be used across (or easily converted for) different defeasible reasoning formalisms.

This would enable the possibility of comparing the *practical* performance of different formalisms which, again, would be highly beneficial to the defeasible reasoning community.

For this to happen, three main issues need to be addressed: firstly, one needs to establish a comparison of current defeasible reasoning formalisms in terms of expressive power. Once the expressivity relationships between formalisms is clear, one can isolate the "intersection" of this expressive power. That is, what kinds of knowledge can *all* of these formalisms express.

The second issue is to identify how to represent the same piece of knowledge in each formalism (i.e., how to translate between different representations). This would pave the way for establishing benchmark ontologies.

Finally, to compare the performances of the different formalisms, one would also need to provide up-to-date implementations for each of them. We have done our part in this thesis by providing one for the preferential approach in DLs.

Chapter 9

Bibliography

[1] Grigoris. Antoniou. A Nonmonotonic Rule System Using Ontologies. In *Proceedings of the International Workshop on Rule Markup Languages for Business Rules on the Semantic Web, held at the International Semantic Web Conference*. Citeseer, 2002.

[2] Grigoris Antoniou and David Billington. Relating Defeasible and Default Logic. In *Proceedings of the Australian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence*, pages 13–24. Springer Berlin Heidelberg, 2001.

[3] Grigoris Antoniou, David Billington, Guido Governatori, and Michael J Maher. Representation Results for Defeasible Logic. *ACM Transactions on Computational Logic (TOCL)*, 2(2):255–287, 2001.

[4] Grigoris Antoniou, David Billington, Guido Governatori, and Michael J Maher. Embedding Defeasible Logic into Logic Programming. *Theory and Practice of Logic Programming*, 6(06):703–735, 2006.

[5] Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The DL-Lite Family and Relations. *Journal of Artificial Intelligence Research*, 36(1):1–69, 2009.

[6] Michael Ashburner, Catherine A Ball, Judith A Blake, David Botstein, Heather Butler, J Michael Cherry, Allan P Davis, Kara Dolinski, Selina S Dwight, Janan T Eppig, et al. Gene Ontology: Tool for the Unification of Biology. *Nature Genetics*, 25(1):25–29, 2000.

[7] Franz Baader. Least Common Subsumers and Most Specific Concepts in a Description Logic with Existential Restrictions and Terminological Cycles. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 3, pages 319–324. Citeseer, 2003.

[8] Franz Baader. Ontology-Based Monitoring of Dynamic Systems. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*. AAAI press, 2014.

[9] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ Envelope. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 5, pages 364–369, 2005.

[10] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele. Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.

[11] Franz Baader and Bernhard Hollunder. Embedding Defaults into Terminological Knowledge Representation Formalisms. *Journal of Automated Reasoning*, 14(1):149–180, 1995.

[12] Franz Baader and Bernhard Hollunder. Priorities on Defaults with Prerequisites, and their Application in Treating Specificity in Terminological Default Logic. *Journal of Automated Reasoning*, 15(1):41–68, 1995.

[13] Franz Baader, Bernhard Hollunder, Bernhard Nebel, Hans-Jrgen Profitlich, and Enrico Franconi. An Empirical Analysis of Optimization Techniques for Terminological Representation Systems. *Applied Intelligence*, 4(2):109–132, 1994.

[14] Franz Baader and Ralf Küsters. Computing the Least Common Subsumer and the Most Specific Concept in the Presence of Cyclic $\mathcal{ALN}$-Concept Descriptions. In *Proceedings of the Annual German Conference on Artificial Intelligence (KI)*, pages 129–140. Springer, 1998.

[15] Franz Baader, Ralf Küsters, and Ralf Molitor. Computing Least Common Subsumers in Description Logics with Existential Restrictions. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 99, pages 96–101, 1999.

[16] Franz Baader, Carsten Lutz, and Boontawee Suntisrivaraporn. Is Tractable Reasoning in Extensions of the Description Logic $\mathcal{EL}$ Useful in Practice? In *Proceedings of the International Workshop on Methods for Modalities*, volume 450, 2005.

[17] Franz Baader, Carsten Lutz, and Boontawee Suntisrivaraporn. CEL A Polynomial-time Reasoner for Life Science Ontologies. In *Automated Reasoning*, pages 287–291. Springer, 2006.

[18] Franz Baader and Paliath Narendran. Unification of Concept Terms in Description Logics. *Journal of Symbolic Computation*, 31(3):277–305, 2001.

[19] Franz Baader and Rafael Peñaloza. Axiom pinpointing in general tableaux. *Journal of Logic and Computation*, 20(1):5–34, 2010.

[20] Franz Baader and Rafael Peñaloza. Axiom Pinpointing in General Tableaux. *Journal of Logic and Computation*, 20(1):5–34, 2010.

[21] Franz Baader, Baris Sertkaya, and Anni-Yasmin Turhan. Computing the Least Common Subsumer w.r.t. a Background Terminology. *Journal of Applied Logic*, 5(3):392–420, 2007.

[22] Samantha Bail, Matthew Horridge, Bijan Parsia, and Ulrike Sattler. The Justificatory Structure of the NCBO Bioportal Ontologies. In

Proceedings of the International Semantic Web Conference, pages 67–82. Springer, 2011.

[23] Samantha Bail, Bijan Parsia, and Ulrike Sattler. The Logical Diversity of Explanations in OWL Ontologies. In *Proceedings of the ACM International Conference on Information & Knowledge Management*, pages 559–568. ACM, 2013.

[24] Chryssoula Bekiari, Martin Doerr, Carlo Allocca, Julien Barde, and Nikos Minadakis. *MarineTLO Documentation*. Forth.

[25] Tim Berners-Lee, James Hendler, Ora Lassila, et al. The Semantic Web. *Scientific American*, 284(5):28–37, 2001.

[26] Hassan Bezzazi, David C. Makinson, and Ramón Pino Pérez. Beyond Rational Monotony: Some Strong Non-Horn Rules for Nonmonotonic Inference Relations. *Journal of Logic and Computation*, 7(5):605–631, 1997.

[27] Piero A Bonatti, Marco Faella, Carsten Lutz, Luigi Sauro, and Frank Wolter. Decidability of Circumscribed Description Logics Revisited. In *Advances in Knowledge Representation, Logic Programming, and Abstract Argumentation*, pages 112–124. Springer, 2015.

[28] Piero A Bonatti, Marco Faella, and Luigi Sauro. $\mathcal{EL}$ with Default Attributes and Overriding. In *Proceedings of the International Semantic Web Conference*, pages 64–79. Springer, 2010.

[29] Piero A Bonatti, Marco Faella, and Luigi Sauro. Adding Default Attributes to $\mathcal{EL}$++. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2011.

[30] Piero A Bonatti, Marco Faella, and Luigi Sauro. Defeasible Inclusions in Low-complexity DLs. *Journal of Artificial Intelligence Research*, pages 719–764, 2011.

[31] Piero A Bonatti, Marco Faella, and Luigi Sauro. On the Complexity of $\mathcal{EL}$ with Defeasible Inclusions. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 22, pages 762–767. Citeseer, 2011.

[32] Piero A Bonatti, Carsten Lutz, and Frank Wolter. Description Logics with Circumscription. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pages 400–410, 2006.

[33] Piero A Bonatti, Carsten Lutz, and Frank Wolter. The Complexity of Circumscription in DLs. *Journal of Artificial Intelligence Research*, pages 717–773, 2009.

[34] Alex Borgida and David W Etherington. Hierarchical Knowledge Bases and Efficient Disjunctive Reasoning. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pages 33–43. Morgan Kaufmann Publishers Inc., 1989.

[35] Alexander Borgida and Ronald J. Brachman. Conceptual Modeling with Description Logics. In *Description Logic Handbook*, pages 349–372, 2003.

[36] Sebastian Brandt. Polynomial-time Reasoning in a Description Logic with Existential Restrictions, GCI Axioms, and What Else? In *Proceedings of the European Conference on Artificial Intelligence*, volume 16, page 298, 2004.

[37] Stefan Brass. Deduction with Supernormal Defaults. In *Nonmonotonic and Inductive Logic*, pages 153–174. Springer, 1993.

[38] Gerhard Brewka. The Logic of Inheritance in Frame Systems. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 483–488. Citeseer, 1987.

[39] Gerhard Brewka. *Nonmonotonic Reasoning: Logical Foundations of Commonsense*, volume 12. Cambridge University Press., 1991.

[40] Gerhard Brewka. Adding Priorities and Specificity to Default Logic. In *Logics in Artificial Intelligence*, pages 247–260. Springer, 1994.

[41] Katarina Britz, Giovanni Casini, Thomas Meyer, Kody Moodley, and Ivan Varzinczak. Ordered Interpretations and Entailment for Defeasible Description Logics. Technical report, CAIR, CSIR Meraka and UKZN, South Africa, 2013. Available at `http://tinyurl.com/cydd6yy`.

[42] Katarina Britz, Thomas Meyer, and Ivan Varzinczak. Semantic Foundation for Preferential Description Logics. In *Proceedings of the Australasian Joint Conference on Artificial Intelligence*, pages 491–500. Springer, 2011.

[43] Katarina Britz, Thomas Meyer, and Ivan Varzinczak. Preferential Role Restrictions. In *Proceedings of the International Workshop on Description Logics*, volume 1014, pages 93–106, 2013.

[44] Marco Cadoli, Francesco M. Donini, and Marco Schaerf. Closed World Reasoning in Hybrid Systems. In Zbigniew W. Ras, Maria Zemankova, and Mary L. Emrich, editors, *Proceedings of the International Symposium on Methodologies for Intelligent Systems*, pages 474–481. Elsevier, 1990.

[45] Marco Cadoli and Maurizio Lenzerini. The Complexity of Propositional Closed World Reasoning and Circumscription. *Journal of Computer and System Sciences*, 48(2):255–310, 1994.

[46] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, Daniele Nardi, and Riccardo Rosati. Description Logic Framework for Information Integration. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pages 2–13, 1998.

[47] Diego Calvanese, Maurizio Lenzerini, and Daniele Nardi. Description Logics for Conceptual Data Modeling. In *Logics for Databases and Information Systems*, pages 229–263. Springer, 1998.

[48] Giovanni Casini, Thomas Meyer, Kody Moodley, and Riku Nortj. Relevant Closure: A New Form of Defeasible Reasoning for Description Logics. In Eduardo Ferm and Joo Leite, editors, *Logics in Artificial Intelligence*, volume 8761 of *Lecture Notes in Computer Science*, pages 92–106. Springer, 2014.

[49] Giovanni Casini, Thomas Meyer, Kody Moodley, Uli Sattler, and Ivan Varzinczak. Introducing Defeasibility into OWL Ontologies. In *Proceedings of the International Semantic Web Conference*, pages 409–426. Springer, 2015.

[50] Giovanni Casini, Thomas Meyer, Kody Moodley, and Ivan Varzinczak. Nonmonotonic Reasoning in Description Logics. Rational Closure for the ABox. In *Proceedings of the International Workshop on Description Logics*, pages 600–615, 2013.

[51] Giovanni Casini and Umberto Straccia. Rational Closure for Defeasible Description Logics. In *Proceedings of the European Conference on Logics in Artificial Intelligence*, pages 77–90, 2010.

[52] Giovanni Casini and Umberto Straccia. Lexicographic Closure for Defeasible Description Logics. In *Proceedings of the Australasian Ontology Workshop*, volume 969, pages 28–39, 2012.

[53] Giovanni Casini, Umberto Straccia, and Thomas Meyer. A Polynomial Time Subsumption Algorithm for $\mathcal{EL}^{\perp}$ under Rational Closure. Technical report, CSIR Meraka Institute, ISTI-CNR, UCT, South Africa and Italy, 2015. Available at `http://tinyurl.com/qx9sdxd`.

[54] Keith L Clark. Negation as Failure. In *Logic and Data bases*, pages 293–322. Springer, 1978.

[55] William W Cohen, Alexander Borgida, Haym Hirsh, et al. Computing Least Common Subsumers in Description Logics. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 754–760, 1992.

[56] William W Cohen and Haym Hirsh. Learning the CLASSIC Description Logic: Theoretical and Experimental Results. *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pages 121–133, 1994.

[57] Roger A Cote and Stanley Robboy. Progress in Medical Information Management: Systematized Nomenclature of Medicine (SNOMED). *Jama*, 243(8):756–762, 1980.

[58] Giuseppe De Giacomo and Maurizio Lenzerini. Boosting the Correspondence Between Description Logics and Propositional Dynamic Logics. In *Proceedings of the National Conference on Artificial Intelligence*, pages 205–212, 1994.

[59] Maarten De Rijke and Holger Sturm. Global Definability in Basic Modal Logic. *Essays on Non-classical Logic*, 1:111–135, 2001.

[60] James P. Delgrande, Torsten Schaub, and W. Ken Jackson. Alternative Approaches to Default Logic. *Artificial Intelligence*, 70(1):167–237, 1994.

[61] Julian Dolby, Achille Fokoue, Aditya Kalyanpur, Li Ma, Edith Schonberg, Kavitha Srinivas, and Xingzhi Sun. *Scalable Grounded Conjunctive Query Evaluation Over Large and Expressive Knowledge Bases*. Springer, 2008.

[62] Francesco M Donini, Maurizio Lenzerini, Daniele Nardi, Andrea Schaerf, and Werner Nutt. Adding Epistemic Operators to Concept Languages. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pages 342–353. Citeseer, 1992.

[63] Francesco M Donini, Daniele Nardi, and Riccardo Rosati. Description Logics of Minimal Knowledge and Negation as Failure. *ACM Transactions on Computational Logic (TOCL)*, 3(2):177–225, 2002.

[64] Kevin Donnelly. SNOMED-CT: The Advanced Terminology and Coding System for eHealth. *Studies in Health Technology and Informatics*, 121:279–290, 2006.

[65] Thomas Eiter, Georg Gottlob, and Yuri Gurevich. Curb Your Theory! A Circumscriptive Approach for Inclusive Interpretation of Disjunctive Information. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 13, pages 634–634, 1993.

[66] David W Etherington. A Semantics for Default Logic. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 87, pages 495–498, 1987.

[67] Enrico Franconi and Gary Ng. The i.com Tool for Intelligent Conceptual Modelling. In *Proceedings of the International Workshop on Knowledge Representation meets Databases*, 2000.

[68] Peter Gärdenfors. *Belief Revision*, volume 29. Cambridge University Press, 2003.

[69] G. Gentzen. Untersuchungen über das logische Schließen. I. *Mathematische Zeitschrift*, 39(1):176–210, 1935.

[70] Laura Giordano, Valentina Gliozzi, Nicola Olivetti, and Gian Luca Pozzato. Reasoning about Typicality in Low Complexity DLs: The Logics $\mathcal{EL}^{\perp}T_{min}$ and $DL\text{-}Lite_cT_{min}$. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 22, pages 894–899. Citeseer, 2011.

[71] Laura Giordano, Valentina Gliozzi, Nicola Olivetti, and Gian Luca Pozzato. Semantic Characterization of Rational closure: From Propositional Logic to Description Logics. *Artificial Intelligence*.

[72] Laura Giordano, Valentina Gliozzi, Nicola Olivetti, and Gian Luca Pozzato. Reasoning about Typicality in Preferential Description Logics. In *Logics in Artificial Intelligence*, pages 192–205. Springer, 2008.

[73] Laura Giordano, Valentina Gliozzi, Nicola Olivetti, and Gian Luca Pozzato. Prototypical Reasoning with Low Complexity Description Logics: Preliminary Results. In *Logic Programming and Nonmonotonic Reasoning*, pages 430–436. Springer, 2009.

[74] Laura Giordano, Valentina Gliozzi, Nicola Olivetti, and Gian Luca Pozzato. Rational Closure in $\mathcal{SHIQ}$. In *Proceedings of the International Workshop on Description Logics*, pages 543–555, 2014.

[75] Laura Giordano, Nicola Olivetti, Valentina Gliozzi, and Gian Luca Pozzato. *mathcalALC* + T: a Preferential Extension of Description Logics. *Fundamenta Informaticae*, 96(3):341–372, 2009.

[76] Birte Glimm, Ian Horrocks, Boris Motik, Rob Shearer, and Giorgos Stoilos. A Novel Approach to Ontology Classification. *Web Semantics: Science, Services and Agents on the World Wide Web*, 14(0):84–101, 2012.

[77] Birte Glimm, Ian Horrocks, Boris Motik, and Giorgos Stoilos. Optimising Ontology Classification. In *Proceedings of the International Semantic Web Conference*, pages 225–240. Springer, 2010.

[78] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. HermiT: an OWL 2 Reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[79] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. HermiT: an OWL 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[80] Francois Goasdoue, Veronique Lattes, and Marie-Christine Rousset. The Use of CARIN Language and Algorithms for Information Integration: The Picsel System. *International Journal of Cooperative Information Systems*, 9(04):383–401, 2000.

[81] Guido Governatori. Defeasible Description Logics. In *Rules and Rule Markup Languages for the Semantic Web*, pages 98–112. Springer, 2004.

[82] Guido Governatori, Michael J Maher, Grigoris Antoniou, and David Billington. Argumentation Semantics for Defeasible Logic. *Journal of Logic and Computation*, 14(5):675–702, 2004.

[83] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Extracting Modules from Ontologies: A Logic-based Approach. In *Modular Ontologies*, pages 159–186. Springer, 2009.

[84] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. OWL 2: The Next Step for OWL. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(4):309–322, 2008.

[85] Stephan Grimm and Pascal Hitzler. Semantic Matchmaking of Web Resources with Local Closed-World Reasoning. *International Journal of Electronic Commerce*, 12(2):89–126, 2007.

[86] Stephan Grimm and Pascal Hitzler. A Preferential Tableaux Calculus for Circumscriptive $\mathcal{ALCO}$. In Axel Polleres and Terrance Swift, editors, *Web Reasoning and Rule Systems*, volume 5837 of *Lecture Notes in Computer Science*, pages 40–54. Springer, 2009.

[87] Benjamin N Grosof, Ian Horrocks, Raphael Volz, and Stefan Decker. Description Logic Programs: Combining Logic Programs with Description Logic. In *Proceedings of the International Conference on World Wide Web*, pages 48–57. ACM, 2003.

[88] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. LUBM: A Benchmark for OWL Knowledge Base Systems. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(2):158–182, 2005.

[89] Volker Haarslev and Ralf Möller. High Performance Reasoning with Very Large Knowledge Bases: A Practical Case Study. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 1, pages 161–168, 2001.

[90] Volker Haarslev and Ralf Müller. RACER System Description. In *Automated Reasoning*, pages 701–705. Springer, 2001.

[91] J. Herbrand. *Recherches sur la theorie de la demonstration*. PhD thesis, University of Paris, 1930.

[92] Stijn Heymans, Li Ma, Darko Anicic, Zhilei Ma, Nathalie Steinmetz, Yue Pan, Jing Mei, Achille Fokoue, Aditya Kalyanpur, Aaron Kershenbaum, et al. Ontology Reasoning With Large Data Repositories. In *Ontology Management*, pages 89–128. Springer, 2008.

[93] Robert Hoehndorf, Frank Loebe, Janet Kelso, and Heinrich Herre. Representing Default Knowledge in Biomedical Ontologies: Application to the Integration of Anatomy and Phenotype Ontologies. *BMC Bioinformatics*, 8(1):377, 2007.

[94] Matthew Horridge. *Justification Based Explanation in Ontologies*. PhD thesis, University of Manchester, 2011.

[95] Matthew Horridge and Sean Bechhofer. The OWL API: A Java API for OWL Ontologies. *Semantic Web*, 2(1):11–21, 2011.

[96] Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Laconic and Precise Justifications in OWL. In *Proceedings of the International Semantic Web Conference*, pages 323–338. Springer, 2008.

[97] Matthew Horridge, Tania Tudorache, Jennifer Vendetti, Csongor I. Nyulas, Mark A. Musen, and Natalya F. Noy. Simplified OWL Ontology Editing for the Web: Is WebProtégé Enough? In Harith Alani, Lalana Kagal, Achille Fokoue, Paul Groth, Chris Biemann, JosianeXavier Parreira, Lora Aroyo, Natasha Noy, Chris Welty, and Krzysztof Janowicz, editors, *Proceedings of the International Semantic Web Conference*, volume 8218 of *Lecture Notes in Computer Science*, pages 200–215. Springer Berlin Heidelberg, 2013.

[98] Ian Horrocks. The Fact System. In *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 307–312. Springer, 1998.

[99] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The Even More Irresistible $\mathcal{SROIQ}$. In *Proceedings of the International Conference on Principles in Knowledge Representation and Reasoning*, volume 6, pages 57–67, 2006.

[100] Ian Horrocks and Peter F Patel-Schneider. Reducing OWL Entailment to Description Logic Satisfiability. In *Proceedings of the International Semantic Web Conference*, pages 17–29. Springer, 2003.

[101] Ian Horrocks, Peter F Patel-Schneider, and Frank Van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The Making of a Web Ontology Language. *Web Semantics: Science, Services and Agents on the World Wide Web*, 1(1):7–26, 2003.

[102] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical Reasoning for Expressive Description Logics. In *Logic for Programming and Automated Reasoning*, pages 161–180. Springer, 1999.

[103] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical Reasoning for Very Expressive Description Logics. *Logic Journal of the IGPL*, 8(3):239–263, 2000.

[104] Aditya Kalyanpur. *Debugging and Repair of OWL Ontologies*. PhD thesis, 2006.

[105] Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. Finding All Justifications of OWL DL Entailments. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudr-Mauroux, editors, *Proceedings of the International Semantic Web Conference*, volume 4825 of *Lecture Notes in Computer Science*, pages 267–280. Springer Berlin Heidelberg, 2007.

[106] Yevgeny Kazakov, Markus Krötzsch, and František Simančík. The Incredible ELK. *Journal of Automated Reasoning*, 53(1):1–61, 2014.

[107] Peihong Ke and Uli Sattler. Next Steps for Description Logics of Minimal Knowledge and Negation as Failure, 2008.

[108] Vladimir Kolovski, Bijan Parsia, and Yarden Katz. Implementing OWL Defaults. In *Proceedings of the OWL Experiences and Directions Workshop*, volume 216, 2006.

[109] Boris Konev, Carsten Lutz, Dirk Walther, and Frank Wolter. Model-theoretic Inseparability and Modularity of Description Logic Ontologies. *Artificial Intelligence*, 203:66–103, 2013.

[110] Kurt Konolige. On the Relation between Default and Autoepistemic Logic. *Artificial Intelligence*, 35(3):343–382, 1988.

[111] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. Nonmonotonic Reasoning, Preferential Models and Cumulative Logics. *Artificial Intelligence*, 44(1–2):167–207, 1990.

[112] Petr Křemen and Zdeněk Kouba. Incremental Approach to Error Explanations in Ontologies. In *Networked Knowledge-Networked Media*, pages 171–185. Springer, 2009.

[113] Ralf Küsters. *Non-standard Inferences in Description Logics*. Springer-Verlag, 2001.

[114] Ora Lassila and Deborah McGuinness. The Role of Frame-based Representation on the Semantic Web. *Linköping Electronic Articles in Computer and Information Science*, 6(5), 2001.

[115] Michael J Lawley and Cyril Bousquet. Fast Classification in Protégé: Snorocket as an OWL 2 EL Reasoner. In *Proceedings of the Australasian Ontology Workshop*, volume 122, pages 45–49, 2010.

[116] Daniel. Lehmann. Another Perspective on Default Reasoning. *Annals of Mathematics and Artificial Intelligence*, 15(1):61–82, 1995.

[117] Daniel Lehmann and Menachem Magidor. What Does a Conditional Knowledge Base Entail? *Artificial Intelligence*, 55(1):1–60, 1992.

[118] Daniel Lehmann, Menachem Magidor, and Karl Schlechta. Distance Semantics for Belief Revision. *The Journal of Symbolic Logic*, 66(01):295–317, 2001.

[119] Hector J. Levesque. Foundations of a Functional Approach to Knowledge Representation. *Artificial Intelligence*, 23(2):155–212, 1984.

[120] Hector J Levesque. Making Believers out of Computers. *Artificial Intelligence*, 30(1):81–108, 1986.

[121] Vladimir Lifschitz. Computing Circumscription. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 85, pages 121–127, 1985.

[122] Vladimir Lifschitz. On Open Defaults. In John W. Lloyd, editor, *Computational Logic*, ESPRIT Basic Research Series, pages 80–95. Springer Berlin Heidelberg, 1990.

[123] Vladimir Lifschitz. Nonmonotonic Databases and Epistemic Queries. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 381–386, 1991.

[124] Vladimir Lifschitz. Circumscription. In *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3, pages 297–352. 1994.

[125] Vladimir Lifschitz. Minimal Belief and Negation as Failure. *Artificial Intelligence*, 70(1):53–72, 1994.

[126] Witold Lukaszewlcz. Two Results on Default Logic. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 1, pages 459–461. Morgan Kaufmann Publishers Inc., 1985.

[127] Carsten Lutz. *The Complexity of Description Logics with Concrete Domains*. PhD thesis, 2002.

[128] Carsten Lutz, Robert Piro, and Frank Wolter. Description Logic TBoxes: Model-theoretic Characterizations and Rewritability. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 983–988. AAAI Press, 2011.

[129] Li Ma, Yang Yang, Zhaoming Qiu, Guotong Xie, Yue Pan, and Shengping Liu. Towards a Complete OWL Ontology Benchmark. In *The Semantic Web: Research and Applications*, volume 4011 of *Lecture Notes in Computer Science*, pages 125–139. Springer Berlin, 2006.

[130] M. J. Maher. A Model-Theoretic Semantics for Defeasible Logic. In *Proceedings of the Workshop on Paraconsistent Computational Logic*, pages 67–80, 2002.

[131] Michael J Maher, Andrew Rock, Grigoris Antoniou, David Billington, and Tristan Miller. Efficient Defeasible Reasoning Systems. *International Journal on Artificial Intelligence Tools*, 10(4):483–501, 2001.

[132] Nicolas Matentzoglu, Samantha Bail, and Bijan Parsia. A Corpus of OWL DL Ontologies. In *Proceedings of the International Workshop on Description Logics*, pages 829–841, 2013.

[133] Nicolas Matentzoglu, Samantha Bail, and Bijan Parsia. A Corpus of OWL DL Ontologies. In *Proceedings of the International Workshop on Description Logics*, pages 829–841, 2013.

[134] Nicolas Matentzoglu, Bijan Parsia, and Uli Sattler. An Empirical Investigation of Difficulty of Subsets of Description Logic Ontologies. In *Proceedings of the International Workshop on Description Logics*, pages 659–670, 2014.

[135] Nicolas Matentzoglu, Daniel Tang, Bijan Parsia, and Uli Sattler. The Manchester OWL Repository: System Description. In *Proceedings of the International Semantic Web Conference*, pages 285–288, 2014.

[136] John McCarthy. Circumscription - A Form of Non-Monotonic Reasoning. *Artificial Intelligence*, 13(1–2):27–39, 1980.

[137] John McCarthy. Applications of Circumscription to Formalizing Common-Sense Knowledge. *Artificial Intelligence*, 28(1):89–116, 1986.

[138] Deborah L McGuinness and Jon R. Wright. Conceptual Modelling for Configuration: A Description Logic-based Approach. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 12(4):333–344, 1998.

[139] Thomas Meyer, Kevin Lee, and Richard Booth. Knowledge Integration for Description Logics. In *Proceedings of the National Conference on Artificial Intelligence*, volume 2, pages 645–650. AAAI Press, 2005.

[140] Thomas Meyer, Kevin Lee, Richard Booth, and Jeff Z Pan. Finding Maximally Satisfiable Terminologies for the Description Logic $\mathcal{ALC}$. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 269. AAAI Press., 2006.

[141] Robert C. Moore. Semantical Considerations on Nonmonotonic Logic. *Artificial Intelligence*, 25(1):75–4, 1985.

[142] Robert C. Moore. Semantical considerations on nonmonotonic logic. *Artificial Intelligence*, 25(1):75–94, 1985.

[143] Nishal A. Morar. Extending Classical Reasoning for Classification Queries Over Ontologies. Master's thesis, University of KwaZulu-Natal, 2015.

[144] Boris Motik and Ian Horrocks. *OWL Datatypes: Design and Implementation*. Springer, 2008.

[145] Bernhard Nebel. Syntax-based Approaches to Belief Revision. *Belief revision*, 29:52–88, 1992.

[146] Nils J. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28(1):71–87, 1986.

[147] Natalya F Noy. Semantic Integration: A Survey of Ontology-based Approaches. *Association for Computing Machinery, Special Interest Group on Management of Data Record*, 33(4):65–70, 2004.

[148] Natalya F. Noy, Nigam H. Shah, Patricia L. Whetzel, Benjamin Dai, Michael Dorf, Nicholas Griffith, Clement Jonquet, Daniel L. Rubin, Margaret-Anne Storey, Christopher G. Chute, and Mark A. Musen. BioPortal: Ontologies and Integrated Data Resources at the Click of a Mouse. *Nucleic Acids Research*, 37(suppl_2):W170–W173, 2009.

[149] Donald Nute. Defeasible logic. *Handbook of Logic in Artificial Intelligence and Logic Programming*, 3:353–395, 1994.

[150] Lin Padgham and Patrick Lambrix. A Framework for Part-of Hierarchies in Terminological Logics. In *Principals of Knowledge Representation and Reasoning*, pages 485–496, 1994.

[151] Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Debugging OWL ontologies. In *Proceedings of the International Conference on the World Wide Web*, pages 633–640. ACM, 2005.

[152] Antonella Poggi, Mariano Rodriguez, and Marco Ruzzi. Ontology-based Database Access With DIG-Mastro and the OBDA Plugin for Protégé. In *Proceedings of the OWL Experiences and Directions Workshop*, volume 8, 2008.

[153] John L Pollock. Defeasible Reasoning. *Cognitive science*, 11(4):481–518, 1987.

[154] David Poole. Variables in hypothesis. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 905–908, 1987.

[155] Pakornpong Pothipruk and Guido Governatori. $\mathcal{ALE}$ defeasible description logic. In *Proceedings of the National Conference on Artificial Intelligence: Advances in Artificial Intelligence*, pages 110–119. Springer, 2006.

[156] Guilin Qi and Jianfeng Du. Model-based Revision Operators for Terminologies in Description Logics. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 891–897. Citeseer, 2009.

[157] Robert G. Raskin and Michael J. Pan. Knowledge Representation in the Semantic Web for Earth and Environmental Terminology (SWEET). *Computers & Geosciences*, 31(9):1119–1125, 2005.

[158] Alan L. Rector. Defaults, Context, and Knowledge: Alternatives for OWL-indexed Knowledge Bases. In *Pacific Symposium on Biocomputing*, volume 9, pages 226–237. World Scientific, 2004.

[159] Alan L Rector, Jeremy E Rogers, and Pam Pole. The GALEN High Level Ontology. *Studies in Health Technology and Informatics*, pages 174–178, 1996.

[160] Raymond Reiter. On Closed World Databases. In Hervé Gallaire and Jack Minker, editors, *Logic and Data Bases*, pages 119–140. Plenum Press, 1978.

[161] Raymond Reiter. A Logic for Default Reasoning. *Artificial Intelligence*, 13(1):81–132, 1980.

[162] Raymond Reiter. A Theory of Diagnosis from First Principles. *Artificial intelligence*, 32(1):57–95, 1987.

[163] Raymond Reiter. On Asking What a Database Knows. In JohnW. Lloyd, editor, *Computational Logic*, ESPRIT Basic Research Series, pages 96–113. Springer Berlin Heidelberg, 1990.

[164] Raymond Reiter. In *Talk given at the Christian Doppler Lab for Expert Systems*, March 31, 1992.

[165] Raymond Reiter and Giovanni Criscuolo. On Interacting Defaults. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 81, pages 270–276, 1981.

[166] Jeremy E Rogers and Alan L Rector. The GALEN Ontology. *Medical Informatics Europe*, pages 174–178, 1996.

[167] Ana A. Romero, Bernardo Cuenca Grau, and Ian Horrocks. MORe: Modular Combination of OWL Reasoners for Ontology Classification. In *Proceedings of the International Semantic Web Conference*, pages 1–16. 2012.

[168] Sebastian Rudolph. Foundations of Description Logics. In Axel Polleres, Claudia d'Amato, Marcelo Arenas, Siegfried Handschuh, Paula Kroner, Sascha Ossowski, and Peter F. Patel-Schneider, editors, *Reasoning Web. Semantic Technologies for the Web of Data – 7th International Summer School.*

[169] Uli Sattler, Thomas Schneider, and Michael Zakharyaschev. Which Kind of Module Should I Extract? In *Proceedings of the International Workshop on Description Logics*, 2009.

[170] Ulrike Sattler, Thomas Schneider, and Michael Zakharyaschev. Which Kind of Module Should I Extract? *International Workshop on Description Logics*, 477:78, 2009.

[171] Viachaslau Sazonau, Ulrike. Sattler, and Gavin Brown. Predicting OWL Reasoners: Locally or Globally? In *International Workshop on Description Logics*, pages 713–724, 2014.

[172] Klaus Schild. A Correspondence Theory for Terminological Logics: Preliminary Report. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 466–471, 1991.

[173] Klaus Schild. *Querying Knowledge and Data Bases by a Universal Description Logic with Recursion.* PhD thesis, 1995.

[174] Stefan Schlobach and Ronald Cornet. Non-standard Reasoning Services for the Debugging of Description Logic Terminologies. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 3, pages 355–362, 2003.

[175] Manfred Schmidt-Schauß and Gert Smolka. Attributive Concept Descriptions with Complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[176] Stefan Schulz, Holger Stenzhorn, Martin Boeker, and Barry Smith. Strengths and Limitations of Formal Ontologies in the Biomedical Domain. *Revista Electronica de Comunicacao, Informacao & Inovacao em Saude: RECIIS*, 3(1):31, 2009.

[177] Kunal Sengupta, Pascal Hitzler, and Krzysztof Janowicz. Revisiting Default Description Logics and Their Role in Aligning Ontologies. In Thepchai Supnithi, Takahira Yamaguchi, Jeff Z. Pan, Vilas Wuwongse,

and Marut Buranarach, editors, *Semantic Technology*, volume 8943 of *Lecture Notes in Computer Science*, pages 3–18. Springer International Publishing, 2015.

[178] Kunal Sengupta, Adila Alfa Krisnadhi, and Pascal Hitzler. Local Closed World Semantics: Grounded Circumscription for OWL. In *Proceedings of the International Semantic Web Conference*, pages 617–632. Springer, 2011.

[179] Rob Shearer and Ian Horrocks. *Exploiting Partial Information in Taxonomy Construction*. Springer, 2009.

[180] Yoav Shoham. Readings in nonmonotonic reasoning. chapter A Semantical Approach to Nonmonotonic Logics, pages 227–250. Morgan Kaufmann Publishers Inc., 1987.

[181] Nicholas Sioutos, Sherri de Coronado, Margaret W. Haber, Frank W. Hartel, Wen-Ling Shaiu, and Lawrence W. Wright. NCI Thesaurus: A Semantic Model Integrating Cancer-related Clinical and Molecular Information. *Journal of Biomedical Informatics*, 40(1):30–43, 2007.

[182] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A Practical OWL-DL Reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):51–53, 2007.

[183] Kent Spackman. Managing Clinical Terminology Hierarchies Using Algorithmic Calculation of Subsumption: Experience with SNOMED-RT. *Journal of the American Medical Informatics Association*, 2000.

[184] Kent A Spackman, Keith E Campbell, and Roger A. Côté. SNOMED RT: A Reference Terminology for Health Care. In *Proceedings of the AMIA Annual Fall Symposium*, page 640. American Medical Informatics Association, 1997.

[185] Andreas Steigmiller, Thorsten Liebig, and Birte Glimm. Konclude: System Description. *Web Semantics: Science, Services and Agents on the World Wide Web*, 27:78–85, 2014.

[186] Robert Stevens, Mikel Egaña Aranguren, Katy Wolstencroft, Ulrike Sattler, Nick Drummond, Matthew Horridge, and Alan Rector. Using OWL to Model Biological Knowledge. *International Journal of Human-Computer Studies*, 65(7):583–594, 2007.

[187] Umberto Straccia. Reasoning within Fuzzy Description Logics. *Journal of Artificial Intelligence Research*, 14:137–166, 2001.

[188] Boontawee Suntisrivaraporn. *Polynomial-Time Reasoning Support for Design and Maintenance of Large-Scale Biomedical Ontologies*. PhD thesis, 2009.

[189] A. Tarski. Über einige fundamentale begrie der metamathematik. In *Classe III*, volume 23, pages 22–29. Proc. of society of sciences and letters Warsaw, 1930.

[190] Edward Thomas, Jeff Z. Pan, and Yuan Ren. TrOWL: Tractable OWL 2 Reasoning Infrastructure. In *The Semantic Web: Research and Applications*, pages 431–435. Springer, 2010.

[191] Dmitry Tsarkov and Ian Horrocks. FaCT++ Description Logic Reasoner: System Description. In *Automated Reasoning*, pages 292–297. Springer, 2006.

[192] Dmitry Tsarkov and Ian Horrocks. FaCT++ Description Logic Reasoner: System Description. In *Automated Reasoning*, pages 292–297. Springer, 2006.

[193] Dmitry Tsarkov and Ignazio Palmisano. Chainsaw: a Metareasoner for Large Ontologies. In *Proceedings of the OWL Reasoner Evaluation Workshop*, 2012.

[194] Yannis Tzitzikas, Carlo Allocca, Chryssoula Bekiari, Yannis Marketakis, Pavlos Fafalios, Martin Doerr, Nikos Minadakis, Theodore Patkos, and Leonardo Candela. Integrating Heterogeneous and Distributed Information about Marine Species through a Top Level Ontology. In Emmanouel Garoufallou and Jane Greenberg, editors, *Metadata and Semantics Research*, volume 390 of *Communications in Computer and Information Science*, pages 289–301. Springer International Publishing, 2013.

[195] Kewen Wang, David Billington, Jeff Blee, and Grigoris Antoniou. Combining Description Logic and Defeasible Logic for the Semantic Web. In Grigoris Antoniou and Harold Boley, editors, *Rules and Rule Markup Languages for the Semantic Web*, volume 3323 of *Lecture Notes in Computer Science*, pages 170–181. Springer Berlin Heidelberg, 2004.

[196] Patricia. L. Whetzel, Natalya F. Noy, Nigam H. Shah, Paul R. Alexander, Csongor Nyulas, Tania Tudorache, and Mark A. Musen. BioPortal: Enhanced Functionality via New Web Services from the National Center for Biomedical Ontology to Access and Use Ontologies in Software Applications. *Nucleic Acids Research*, 39(suppl_2):W541–W545, 2011.