PRACTICAL UNIFORM INTERPOLATION FOR EXPRESSIVE DESCRIPTION LOGICS

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Engineering and Physical Sciences

2015

Patrick Koopmann
School of Computer Science

Contents

1 Introduction **15**

1.1 Applications of Uniform Interpolation 16

1.2 Challenges and Contributions 20

1.3 Overview of the Thesis 21

1.4 Published Results 22

2 Related Work **24**

2.1 Forgetting in Classical Logics 24

2.2 Second-Order Quantifier Elimination 26

2.3 Uniform Interpolation in Modal Logics 28

2.4 Uniform Interpolation in DL-Lite and $\mathcal{EL}$ 30

2.5 Uniform Interpolation in $\mathcal{ALC}$ 31

2.6 Related Problems in Description Logics 34

2.7 Saturation-Based Reasoning 36

3 Basics of Description Logics and Uniform Interpolation **40**

3.1 Description Logics 40

3.2 Uniform Interpolation 46

3.3 Fixpoints 47

4 Practical Uniform Interpolation for $\mathcal{ALC}$ **49**

4.1 The Normal Form 51

4.2 The Refutation Calculus 58

4.3 Refinements and Redundancy Elimination 79

4.4 The Interpolation Procedure 84

4.5 Examples 99

5 Extending the Method to $\mathcal{SH}$ and $\mathcal{SIF}$ 103

5.1 Role Hierarchies . . . 105

5.2 Transitive Roles . . . 110

5.3 Inverse Roles . . . 116

5.4 Functional Role Restrictions . . . 124

5.5 Bringing It All Together: $\mathcal{SIF}$ . . . 138

5.6 Refutational Completeness of $Res^{s}_{\mathcal{SIF}}$ . . . 141

6 Uniform Interpolation with Cardinality Restrictions 154

6.1 The Refutation Calculus . . . 157

6.2 Refutational Completeness of $Res_{\mathcal{SHQ}}$ . . . 165

6.3 The Interpolation Calculus . . . 175

6.4 Interpolation Completeness of $Int_{\mathcal{SHQ}}$ . . . 178

6.5 Examples . . . 182

7 Uniform Interpolation with ABoxes 186

7.1 The Calculus . . . 191

7.2 Refutational Completeness . . . 195

7.3 Representing the Result in $\mathcal{SHOI}$ . . . 202

8 Implementation and Evaluation 206

8.1 Implementation . . . 208

8.2 The Corpus . . . 215

8.3 Forgetting Few Symbols . . . 217

8.4 Uniform Interpolants for Small Signatures . . . 220

8.5 Uniform Interpolants for Central Signatures . . . 225

9 Conclusion and Future Directions 230

9.1 Future Directions . . . 232

Bibliography 235

Word count: 75,515

List of Tables

3.1 Description logic families and their extensions. 43

3.2 Interpretation of roles and concepts. 44

3.3 Translation from description logic expressions to first-order logic. 45

8.1 Statistics about the corpus used. 217

8.2 Forgetting 50 and 100 symbols with the prototypes. 218

8.3 Forgetting concept and role symbols exclusively. 220

8.4 Results of computing uniform interpolants for small signature sizes. . . 221

8.5 Results for computing $\mathcal{ALCH}$ uniform interpolants for signatures of size 50, selected based on two different heuristics. 228

9.1 Overview of the calculi developed in the thesis. 231

List of Figures

4.1 Transformation rules transforming any $\mathcal{ALC}\nu$ ontology into $\mathcal{ALC}$ normal form. 52

4.2 Rewrite rules to eliminate definers from normal clause sets. 55

4.3 Rules of calculus $Res_{\mathcal{ALC}}$. 59

4.4 Venn diagram illustrating $\models (C_1 \sqcup C_2) \sqcap (C_3 \sqcup C_4) \sqsubseteq C_1 \sqcup C_3 \sqcup (C_2 \sqcap C_4)$. 60

4.5 Saturation $\mathcal{N}_3^*$ of the clausal representation $\mathcal{N}_3$ of the satisfiable example ontology $\mathcal{O}_3$. 67

4.6 The enumerated set $\mathcal{N}_3^*$, ordered by $\prec_c$. 70

4.7 The candidate model $\mathcal{I}_3$ built throughout Examples 4.2.8 and 4.2.18. . 77

4.8 Simplification rules. 83

4.9 Additional rule for the interpolation complete calculus $Int_{\mathcal{ALC}}$. 85

5.1 Additional inference rules in $Res_{\mathcal{ALCH}}$ and $Int_{\mathcal{ALCH}}$. 105

5.2 Modified role propagation rules for $Res^s_{\mathcal{ALCH}}$. 109

5.3 Transitivity rule of $Res_{\mathcal{SH}}$ and $Int_{\mathcal{SH}}$. 112

5.4 Role inversion rule of $Res_{\mathcal{ALCHI}}$ and $Int_{\mathcal{ALCHI}}$. 118

5.5 Model of $\mathcal{O}_{\mathcal{ALCHI}}$, an ontology with inverse roles. 119

5.6 Additional inference rules in $Res_{\mathcal{ALCHF}}$ and $Int_{\mathcal{ALCHF}}$. 126

5.7 Universalisation rule in $Res_{\mathcal{SIF}}$ and $Int_{\mathcal{SIF}}$. 139

5.8 Infinite model of $\mathcal{O}^\infty$. 142

5.9 Model candidate for $\mathcal{N}^*$ generated by method for $Res_{\mathcal{ALC}}$. 143

5.10 Model candidate for $\mathcal{N}^\infty*$ built using the new method. 146

5.11 Interpretation of $\mathcal{O}_{\mathcal{SHIF}}$ that does not satisfy the role hierarchy. 151

5.12 Complete set of rules introduced in this chapter, Part 1. 152

5.13 Complete set of rules introduced in this chapter, Part 2. 153

6.1 Inference rules of the calculus $Res_{\mathcal{SHQ}}$. 159
6.2 Additional inference rules of $Int_{\mathcal{SHQ}}$. 175

7.1 Graph structure of an exemplary ABox interpretation. 189
7.2 Model of a simple non-tree shaped ABox. 190
7.3 Inference rules of the calculi $Res_{\mathcal{SHI}_\vee}$ and $Int_{\mathcal{SHI}_\vee}$. 194

8.1 Role restriction resolution rule used in the implementations. 212
8.2 Logarithmic distribution of symbol frequencies in ontologies of the NCBO BioPortal repository. 223
8.3 Logarithmic distribution genuine modules per symbol in ontologies of the NCBO BioPortal repository. 227

Abstract

Practical Uniform Interpolation for Expressive Description Logics

Patrick Koopmann

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy, 2015

The thesis investigates methods for uniform interpolation in expressive description logics. Description logics are formalisms commonly used to model ontologies. Ontologies store terminological information and are used in a wide range of applications, such as the semantic web, medicine, bio-informatics, software development, data bases and language processing. Uniform interpolation eliminates terms from an ontology such that logical entailments in the remaining language are preserved. The result, the uniform interpolant, is a restricted view of the ontology that can be used for a variety of tasks such as ontology analysis, ontology reuse, ontology evolution and information hiding. Uniform interpolation for description logics has only gained an interest in the research community in the last years, and theoretical results show that it is a hard problem requiring specialised reasoning approaches. We present a range of uniform interpolation methods that can deal with expressive description logics such as $\mathcal{ALC}$ and many of its extensions. For all these logics, these are the first methods that are able to compute uniform interpolants for all inputs. The methods are based a new family of saturation-based reasoning methods, which make it possible to eliminate symbols in a goal-oriented manner. The practicality of this approach is shown by an evaluation on realistic ontologies.

Declaration

No portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright Statement

Acknowledgements

First of all I would like to thank my supervisor Renate A. Schmidt for her guidance, trust, and continuous support throughout the project, and my examiners Ian Pratt-Hartmann and Carsten Lutz for their valuable suggestions on how to improve my thesis. I also thank the Engineering and Physical Sciences Research Council for funding my PhD with an EU Doctoral Training Award. Furthermore, I have to thank Robert Fraile, who gave me the idea of doing a PhD in the United Kingdom, and without whom I would not be here, and Carola Eschenbach from the University of Hamburg, who started my interest in description logics. I also want to thank my office colleagues, who supported me with fruitful discussions on technical and scientific matters, and with whom I shared great moments during my PhD: Mohammad Khodadadi, Roman Krenický, Alexander Constantin, Chiara Del Vescovo, Nico Matentzoglu, Reyadh Alluhaibi, Ayo Adeniyi, Yizheng Zhao, Michał Zawidski, and especially Fabio Papacchini, who introduced me to the delights of an Italian life style and became a good friend of mine. Special thanks also go to Marc, Susie, Curtis and all the Aardvark staff for keeping my brain supplied with caffeine. Last but not least, I have to thank all the new friends I made in Manchester, who made sure I did not get too absorbed by my work, and helped to make these last three years a very special part of my life.

Chapter 1

Introduction

Ontologies are knowledge systems that are used to model terminological knowledge about a particular domain of interest, and are used in areas as diverse as medicine, bio-informatics, the semantic web, software development, and many more. Usually, ontologies define the meaning of and relations between concepts and roles using a description logic, which provides a formal semantic framework and allows automated reasoning systems to infer implicit information from the ontology. Modern applications have lead to the development of very large ontologies such as the medical SNOMED CT (Stearns et al., 2001), which is a component in health information systems in various countries, covering over 300,000 concepts, the National Cancer Institute Thesaurus (NCI) (Sioutos et al., 2007), covering over 100,000 concepts, and the Gene Ontology GO (Gene Ontology Consortium, 2004), covering about 40,000 concepts.

Since a lot of information present in ontologies is implicit, and entries in an ontology can interact in non-trivial ways, understanding and managing a complex ontology requires appropriate tool support. Uniform interpolation methods have the potential to support various ontology related tasks. Uniform interpolation, which also has been studied under the names forgetting, projection, marginalisation and variable elimination, is a means to extract possibly implicit information from an ontology based on a specified set of concept and role symbols, a so-called signature. The result, the uniform interpolant, is a restricted view of that ontology, possibly in a syntactically different form, that only uses the symbols from the specified signature, while preserving all logical entailments of the ontology that are in this signature. This way, it can

be used to extract parts of an ontology for reuse or analysis or to remove confidential information, and has many more applications.

This chapter gives an overview of applications for uniform interpolation, its challenges, and the main contributions of the thesis.

1.1 Applications of Uniform Interpolation

Ontology Summary. Comprehending a complex ontology can be hindered by a too large vocabulary. If the central concepts and roles of the ontology are known, uniform interpolation can be used to compute a more focused high-level summary of the ontology (Lutz and Wolter, 2011; Wang et al., 2010b).

Ontology Reuse. Existing ontologies provide information about a lot of domains for a broad scope. It is therefore reasonable in a lot of contexts to reuse an existing ontology instead of developing a new one from scratch. However, not all information from an existing ontology might be important for a particular application, and importing the complete ontology leads to unnecessary complexity. It is for this reason that methods for automated extraction of sub-ontologies gained a lot of interest in the last years (Grau et al., 2008; Sattler et al., 2009; Konev et al., 2009a; Kontchakov et al., 2010; Nikitina and Glimm, 2012; Gatens et al., 2014; Romero et al., 2015). Module extraction methods compute subsets of the ontology that preserve all entailments over given sets of concepts and roles. However, since modules are syntactical subsets of the original ontology, they may contain much more concepts and roles than are specified. On the other hand, uniform interpolants only use the concept and role symbols that are specified. Uniform interpolation can be used to eliminate symbols from an ontology or module that are not needed for a particular application, so that ontologies more suited for reuse are generated.

Ontology Analysis. With increasing complexity, ontologies become harder to manage and understand, since the axioms in an ontology can interact in non-trivial ways. For ontology engineers and users it is therefore important to provide convenient and reliable ontology analysis tools. The uniform interpolant for a specified set of symbols of interest is a direct representation of the relations between them. Uniform interpolation has therefore the potential to be a powerful analysis tool for ontology

engineers and users (Lutz and Wolter, 2011; Wang et al., 2010b).

Logical Difference. Tools for computing the difference between text files are indispensable tools for comparing different versions of files, variations of which are used in software development to control the evolution of software products. For ontologies however, the syntactical difference of the textual representation is hardly useful, since the same information can be expressed in different ways and a lot of important information is implicit. A semantic approach, which exhibits differing logical entailments of two ontologies, is much more valuable. This semantic notion of difference is called *logical difference* in the literature (Konev et al., 2008b, 2012). If the signatures of two ontologies differ, one is particularly interested in exhibiting entailments in the common signature, or possibly some subset of it.

There has been a lot of research on algorithms that decide whether two ontologies share the same entailments over given signature and to compute these different entailments (Lutz et al., 2007; Pokrywczynski and Walther, 2008; Lutz and Wolter, 2010; Konev et al., 2012, 2013; Botoeva et al., 2014). Ludwig and Konev (2014) propose uniform interpolation as a means to compute logical differences in expressive description logics. Given the uniform interpolant of the second ontology for the signature of interest, one can check using existing description logic reasoners which axioms of it are logically entailed by the first ontology. The remaining axioms are a representation of the new entailments in the selected signature.

Information Hiding. As pointed out by Grau (2010), ontology-based systems are increasingly used in a range of applications that deal with sensitive information, such as in health care systems. If these data are accessed by different users, it is a critical requirement that confidentiality of private information is preserved, and that users have different access on the data depending on their privileges. Such privileges could for example restrict the visibility of certain concepts and roles. One approach to deal with hidden terms is to restrict the access to the ontology using a black box approach, where reasoners and systems can send queries to an ontology-based system that are only allowed to use a restricted language, based on the privileges of the user. This approach is followed by the import-by-query approach investigated by Grau and Motik (2012). An alternative approach is to share a uniform interpolant that only uses the terms a specific user is allowed to see. This approach is particularly useful

if the owner of an ontology decides to share part of its ontology with other users for reuse, but only wants to disclose limited information.

Ontology Obfuscation. In software engineering, obfuscation transforms a program into a functionally equivalent one that is harder to read and understand by human users, in order to avoid reverse-engineering of the software (Collberg et al., 1998). Uniform interpolation could be used to transfer this idea to the domain of ontologies. Ontologies often contain auxiliary concepts that serve the purpose of structuring the content. These structuring concepts can be regarded as proprietary knowledge, and may not be of importance for the intended application by end-users. By forgetting these concepts, one obtains an ontology that loses this structure and is harder to read and edit by a user, while still preserving its intended functionality (Ludwig and Konev, 2013).

Conflict Resolution in Distributed Ontologies. A central idea of the semantic web is to integrate ontologies from different sources, which are developed and maintained by different parties. A major challenge in managing such a system of distributed and dynamic ontologies is to handle inconsistencies that are introduced when integrating these ontologies. In the context of propositional belief systems, Lang and Marquis (2002) propose a solution to this problem based on forgetting. By forgetting the variables that cause a conflict, one can create a set of belief systems that can be merged without resulting in an inconsistent system. This idea has been followed in Qi et al. (2008) to manage inconsistencies among distributed ontologies using uniform interpolation for description logics. A similar approach is used in Qi and Du (2009), who define a belief revision operation in the spirit of Gärdenfors (1988) on the basis of forgetting.

TBox Abduction. Abduction, originally introduced by Peirce (1878), deals with the problem of finding a rational explanation of an observation in a logic-oriented manner. Formally, given an *observation* O in form of a logical formula, and a theory T of *background knowledge*, in abduction we are interested in computing a logical formula H, the *hypothesis*, such that T and H together logically imply the observation. In order to avoid trivial answers, usually a set of relevance criteria is specified. A typical approach is to specify the relevance conditions via a set of *abducibles*, predicate symbols that are allowed in the hypothesis (Paul, 1993). *TBox abduction* is the

task of explaining TBox axioms based on an ontology of background knowledge. An alternative view is that, given an incomplete ontology and a set of desired entailments for that ontology, TBox abduction computes a set of axioms that have to be added to the ontology in order to ensure the desired entailments are logically entailed. This has applications especially in ontology debugging (Elsenbroich et al., 2006; Bienvenu, 2008).

TBox abduction with abducibles can be reduced to uniform interpolation. Given an ontology $\mathcal{O}$ and a TBox axiom $\alpha_O = C_1 \sqsubseteq C_2$, we can represent the negation of α as an ABox axiom $(C_1 \sqcap \neg C_2)(a)$, where a is a special designated individual name, and compute the uniform interpolant of $\mathcal{O} \cup \{(C_1 \sqcap \neg C_2)(a)\}$ for $\mathcal{S}$, where $\mathcal{S}$ is the set of abducibles allowed in the hypothesis. In the uniform interpolant, we can express all ABox axioms on a in a single ABox axiom $H(a)$. By negating H, we obtain a hypothesis TBox axiom $\alpha_H = \top \sqsubseteq \neg H$ such that $\mathcal{O} \cup \{\alpha_H\} \models \alpha$, which is an answer of the abduction problem.

Approximation. Logical formulae can be approximated into weakest sufficient and strongest necessary conditions (Lin, 2001; Doherty et al., 2001). Given a formula F, a theory T and a set of predicates $\mathcal{S}$, a *strongest necessary condition* of F on $\mathcal{S}$ under T is a formula F_{snc} such that $T \cup F \models F_{snc}$, such that there is no formula F' with $T \cup F \models F'$, $T \cup F' \models F_{snc}$, and $T \cup F_{snc} \not\models F'$. A *weakest sufficient condition* F_{wsc} of F on $\mathcal{S}$ under T is a formula F_{wsc} such that $T \cup F_{wsc} \models F$, and there is no formula F' such that $T \cup F' \models F$, $T \cup F_{wsc} \models F'$ and $T \cup F' \not\models F_{wsc}$. Approximation is for example useful in multi-agent systems in which several agents communicate and only share a restricted vocabulary (Doherty et al., 2004). Strongest necessary and weakest sufficient conditions can provide lower and an upper approximations to queries between agents in such systems.

The strongest necessary condition F_{snc} on $\mathcal{S}$ under T corresponds to the uniform interpolant of $T \cup F$ for $\mathcal{S}$, and the weakest sufficient condition coincides with the abduction hypothesis, which can be reduced to uniform interpolation as described above.

More potential applications of uniform interpolation can be found in Gabbay et al. (2008), which deals with the more general problem of second-order quantifier elimination.

1.2 Challenges and Contributions

Uniform interpolation in description logics has gained interest in the research community only in the last years. An investigation of its theoretical properties showed that it is a potentially hard problem. Uniform interpolants cannot always be represented finitely, and if they can, already for the description logics $\mathcal{EL}$ and $\mathcal{ALC}$, their size can be triple exponential in the size of the input ontology (Lutz and Wolter, 2011; Nikitina and Rudolph, 2014).

This thesis is concerned with the description logic $\mathcal{ALC}$ and more expressive ones, and the question of whether it is possible to develop practical methods for uniform interpolation in these languages that are suited for the envisioned applications. So far, existing methods for uniform interpolation in expressive description logics can in general only compute approximations of the uniform interpolant (see Wang et al., 2014; Lutz and Wolter, 2011; Ludwig and Konev, 2014). In contrast, we developed a solution that uses fixpoint operators, such that a finite and precise representation of the uniform interpolant can always be computed. For applications that require a more classical representation, fixpoints can easily be simulated in classical description logics using auxiliary concept symbols, as we will discuss. Greatest fixpoint operators have been used before for representing uniform interpolants (Lutz et al., 2010; Nikitina, 2011), but only for the less expressive description logic $\mathcal{EL}$.

Due to the worst-case complexity and novelty of the problem, new reasoning methods are required if we want to compute uniform interpolants practically. For this reason, a new family of saturation-based reasoning methods has been developed as part of the project. These methods can be used to compute uniform interpolants in description logics up to the expressivities of $\mathcal{SHI}$, $\mathcal{SIF}$ and $\mathcal{SHQ}$. These are important sublanguages of the web ontology standard languages OWL DL 1.1 and OWL DL 2.0, which additionally support more complex role constructors, nominals and data types.

Furthermore, we show how uniform interpolation with expressive description logics can be applied on knowledge bases with ABoxes, that is, how to eliminate concepts and roles from a knowledge base that not only contains terminological information, but also assertions about individual objects, as it could for example be found in a

database. This makes it possible to use uniform interpolation in applications where data play a major role, for example in information hiding.

The practicality of the developed methods is investigated in an evaluation of implemented prototypes of these methods, showing that they can indeed be used to compute uniform interpolants in a range of realistic use cases.

The main contributions of the thesis are the following:

- We present the first method that can compute a finite uniform interpolant for every $\mathcal{ALC}$ ontology and signature. This is also the first practical method that is able to eliminate role symbols from an ontology.

- We present uniform interpolation methods for 6 additional description logics, which extend $\mathcal{ALC}$ up to the expressivities of $\mathcal{SH}$, $\mathcal{SIF}$ and $\mathcal{SHQ}$, and for which uniform interpolation has not been investigated before.

- We present a method for computing uniform interpolants of $\mathcal{SHI}$ knowledge bases with ABoxes. This is the first method for an expressive description logic that can deal with ABoxes of arbitrary structure, and makes use of a new transformation technique into $\mathcal{SHOI}$.

- All these methods are based on a new family of saturation-based reasoning methods, which use a new technique of introducing symbols dynamically.

- The practicality of these approaches is supported by an evaluation on realistic ontologies. We also provide two simple heuristics for choosing signatures for which uniform interpolants are easy to compute.

1.3 Overview of the Thesis

In Chapter 2, we describe existing methods that have been developed for computing uniform interpolants in description logics and related logics, as well as methods for related problems. In addition, we give an overview of saturation-based reasoning procedures for these logics, since a major part of the thesis deals with the development of new saturation procedures that can be used for uniform interpolation.

Chapter 3 contains the preliminaries, in which we formally introduce description logics, fixpoints and uniform interpolation.

The main calculus for computing uniform interpolants is presented in Chapter 4. This is the first method to compute finite representations of uniform interpolants of $\mathcal{ALC}$ ontologies independent of the input, and it is the first practical uniform interpolation method that can eliminate both concept and role symbols. The method is based on a new saturation-based reasoning calculus $Res_{\mathcal{ALC}}$, which we extend to a method for computing finite representations of $\mathcal{ALC}$ uniform interpolants.

The method serves as a basis for further uniform interpolation methods that can deal with more expressive description logics. In Chapter 5, we extend the calculus and the uniform interpolation method step by step to include more description logic operators, to develop a uniform interpolation method for the description logics $\mathcal{SH}$ and $\mathcal{SIF}$.

Whereas these extensions are implemented by extending the initial calculus for $\mathcal{ALC}$ stepwise by new rules, for the description logic $\mathcal{SHQ}$, we develop a new calculus, whose rules can be seen as a generalisation of the rules of the earlier calculi. This is done in Chapter 6. $\mathcal{SHQ}$ uses so-called cardinality restrictions, which require a more complex and harder to prove approach as the earlier description logics. For this reason, we deal with $\mathcal{SHQ}$ in its own chapter.

All these methods deal with uniform interpolation of ontologies. Chapter 7 shows how our approaches can be extended to uniform interpolation of $\mathcal{SHI}$ knowledge bases that consist of both an ontology and an ABox.

All methods have been implemented and evaluated for various use cases. The results of the evaluation are presented in Chapter 8, where we show that the proposed methods are indeed practical for a lot of applications, but also discuss limitations of the current implementation.

The thesis finishes with a conclusion and an overview of future work in Chapter 9.

1.4 Published Results

Some results presented in this thesis have been published at conferences and workshops. The core method for computing uniform interpolants in $\mathcal{ALC}$ presented in

Chapter 4 was first described at the conference Frontiers of Combining Systems (FroCoS) in 2013 (Koopmann and Schmidt, 2013a), though only for the case of forgetting concept symbols. The method for forgetting concept and role symbols, for the description logic $\mathcal{ALCH}$, as presented in Chapter 4 and Section 5.1, was presented at the conference Logic for Programming, Artificial Intelligence and Reasoning (LPAR) in 2013 (Koopmann and Schmidt, 2013c). The LPAR paper also introduces the role resolution rule presented in the evaluation chapter (Section 8.1). Uniform interpolation for $\mathcal{SIF}$, presented in Sections 5.5 and 5.6, was presented at the Description Logic Workshop in 2015 (Koopmann and Schmidt, 2015b). The method for the description logic $\mathcal{SHQ}$, described in Chapter 6, was presented at the International Joint Conference on Automated Reasoning (IJCAR) in 2014 (Koopmann and Schmidt, 2014a). Uniform interpolation for knowledge bases with ABoxes, which is the topic of Chapter 7, was presented at the Description Logic Workshop in 2014 (Koopmann and Schmidt, 2014c) and at the AAAI Conference in 2015 (Koopmann and Schmidt, 2015a), though only for the description logic $\mathcal{ALC}$. Some of the implementation details presented in Section 8.1 have been described in a publication on practical aspects of uniform interpolation for $\mathcal{ALC}$ at the Workshop on Modular Ontologies (WoMO) in 2013 (Koopmann and Schmidt, 2013b). The implemented methods have been made publicly available in form of the tool and Java library Lethe, which was presented at the OWL Reasoner Evaluation Workshop (ORE) in 2015 (Koopmann and Schmidt, 2015c).

Chapter 2

Related Work

2.1 Forgetting in Classical Logics

In propositional logics, uniform interpolation has been studied and used under the more general names *forgetting* and *variable elimination*. For propositional logics, the result of forgetting a symbol p from a formula F can simply be defined as a formula equivalent to $F[p/\top] \vee F[p/\bot]$. The perhaps oldest mentioning of this form of forgetting is made in Boole (1854). Later on, forgetting played an important role in the first automated reasoning systems under the name *variable elimination*. For example, the famous Davis-Putnam algorithm (Davis and Putnam, 1960) decides satisfiability of a formula by essentially eliminating each propositional variable from the input one after another. Other papers discuss the problem of forgetting in propositional logic for other applications, for example Lang and Marquis (2002), Kohlas et al. (1999) and Lang et al. (2003).

Different normal form representations allow for different techniques to efficiently forget propositional symbols. If a propositional formula is in disjunctive normal form, we can forget a symbol p by removing all unsatisfiable clauses and removing all literals of the form p and $\neg p$ in the remaining clauses (Lang et al., 2003). If it is in conjunctive normal form, forgetting a symbol can be achieved by computing the resolvents on that symbol (as in Davis and Putnam (1960) or Kohlas et al. (1999)).

In first-order logic, most methods for forgetting were studied in the context of second-order quantifier elimination. Since this is a problem in its own right and its motivation is slightly different, we discuss second-order quantifier elimination in detail

in Section 2.2.

Though methods for second-order quantifier elimination have long been studied at this point, Lin and Reiter (1994) first introduced the term "forgetting" to denote the elimination of predicates in first-order logic formulae. Lin and Reiter define forgetting in the following way, which is also referred to as *strong forgetting* by other authors (Zhang and Zhou, 2010):

Definition 2.1.1 (Lin and Reiter 1994)**.** Let F be a first-order logic formula and P a predicate. Then, F' is a *result of strongly forgetting P in F* if for every interpretation M, we have $M \models F'$ iff there is an interpretation M' with $M' \models F$ such that M' agrees with M on every element except P.

Note that this original definition does not explicitly require the result of forgetting P to not contain P, even though this is usually assumed in the literature on forgetting. As the authors note, the result of forgetting a predicate P from a formula F is equivalent to the second-order formulae $\exists X.F[P/X]$. This reduces the problem of forgetting to the problem of second-order quantifier elimination discussed in the next section. It also implies that the result of forgetting cannot always be represented as a first-order formula.

Zhang and Zhou (2010) propose an alternative notion, called *weak forgetting*, which is closer to the definition we use throughout the thesis. The result of weak forgetting is always first-order representable, but possibly only by an infinite set of formulae.

Definition 2.1.2 (Zhang and Zhou 2010)**.** Let F be a first-order logic formula and P a predicate. F' is the result of forgetting P iff for every formula G that does not contain P, we have $F' \models G$ iff $F \models G$.

The result of weak forgetting is not always equivalent to the result of strong forgetting, but if the result of strong forgetting is first-order representable, the notions coincide. For applications that are more concerned with logical entailments than with models, weak forgetting provides a sufficient definition.

Since the result of weak forgetting may be an infinite set of formulae, Zhou and Zhang (2011) investigate the notion of *bounded forgetting*, whose results only preserve logical consequences up to a certain *quantifier rank*. The quantifier rank of a formula

is defined as the maximum depth of nestings of existential or universal quantifiers. For a given quantifier rank the result of bounded forgetting is always finite.

2.2 Second-Order Quantifier Elimination

Second-order quantifier elimination is concerned with the problem of transforming second-order formulae into equivalent first-order logic formulae. This is achieved by eliminating quantified predicate symbols. Due to the equivalence $\forall X.F \equiv \neg\exists X.F$, it is sufficient to focus on eliminating existentially quantified predicates. Therefore, the general problem of second-order quantifier elimination is to transform a formula of the form $\exists\vec{X}.F$ into an equivalent formula F' without existential quantifiers over predicate symbols.

As we observed earlier, second-order quantifier elimination in this sense directly corresponds to strong forgetting in first-order logic, since the result of strongly forgetting a predicate P from a formula F is equivalent to $\exists X.F[P/X]$. Second-order quantifier elimination is highly undecidable (Gabbay et al., 2008; Ackermann, 1935). Nonetheless, successful techniques from the area of second-order quantifier elimination can directly be used for forgetting predicate symbols in first-order logic formulae.

There are in general two approaches to eliminate second-order quantifiers. The first, which is implemented in the SCAN algorithm and in the hierarchical resolution approach (Gabbay and Ohlbach, 1992; Bachmair et al., 1994; Ohlbach, 1996), uses resolution on clausal representations of the input to eliminate quantified variables. The second approach, which is used in the algorithms DLS, DLS*, SQEMA and MSQEL (Szalas, 1993; Doherty et al., 1997; Nonnengart and Szałas, 1995; Conradie et al., 2006; Schmidt, 2012), uses rewrite rules on the input formula until quantified predicates can be eliminated using known equivalences. These methods have been used for a variety of applications, for example in artificial intelligence and for automating correspondence theory of modal logic (a detailed discussion of applications can be found in Gabbay et al., 2008).

Ackermann's lemma, which has been first presented in Ackermann (1935), plays a central role in all these approaches, and states the following:

Lemma 2.2.1 (Ackermann 1935)**.** *Let X be a predicate variable, $F(\vec{x},\vec{y})$ a formula in*

which X does not occur, $G_p[X]$ a formula in which X occurs only positively and $G_n[X]$ a formula in which X occurs only negatively. Then, the following equivalences hold:

$$\exists X.\forall \vec{x}\big(\neg X(\vec{x}) \vee F(\vec{x},\vec{y})\big) \wedge G_p[X] \equiv G_p[X/F(\vec{x},\vec{y})]$$
$$\exists X.\forall \vec{x}\big(X(\vec{x}) \vee F(\vec{x},\vec{y})\big) \wedge G_n[X] \equiv G_n[X/F(\vec{x},\vec{y})],$$

where in $G_p[X/F(\vec{x},\vec{y})]$ and $G_n[X/F(\vec{x},\vec{y})]$, the variables in $\vec{x}$ are renamed to respective arguments of X in G_p and G_n.

If the input formula is represented in clausal form, Ackermann's lemma can be applied by resolving on the symbol to be eliminated. This approach is followed by SCAN and the hierarchical resolution approach (Gabbay and Ohlbach, 1992; Bachmair et al., 1994; Ohlbach, 1996). SCAN is the first algorithm for forgetting in first-order logic, and is used in the first implementations of forgetting beyond propositional logic (Gabbay and Ohlbach, 1992; Engel, 1996).

An alternative approach is followed by the DLS system (Doherty et al., 1997). Using Skolemisation and equivalence preserving transformations, DLS tries to transform input formulae into a form that allows the quantified variables to be eliminated using Ackermann's lemma. Skolemisation (Baaz et al., 2001; Nonnengart and Weidenbach, 2001) is used to eliminate first-order existential quantifiers of variables by introducing function symbols. If the above process manages to eliminate all quantified predicates, in a last step the formula has to be unskolemised by eliminating function symbols and reintroducing corresponding existential quantifiers. Depending on the input formula, the unskolemisation step does not always succeed, but if it does, the result is a formula in first-order logic that is equivalent to the second-order input formula.

In Nonnengart and Szałas (1995), this approach is extended by using the following generalisation of Ackermann's lemma, which is referred to as generalised Ackermann's lemma.

Theorem 2.2.2 (Nonnengart and Szałas 1995)**.** *Let X be a predicate variable, $F_p[X]$ and $G_p[X]$ be formulae in which X occurs only positively and $F_n[X]$ and $G_n[X]$ be formulae in which X occurs only negatively. Additionally, assume X occurs in $G_p[X]$ and $G_n[X]$ only with variables as arguments. Then, the following equivalences hold:*

$$\exists X.\forall \vec{x}\big(\neg X(\vec{x}) \vee F_p[X]\big) \wedge G_p[X] \equiv G_p[X/\nu X(\vec{x}).F_p[X])]$$
$$\exists X.\forall \vec{x}\big(X(\vec{x}) \vee F_n[X]\big) \wedge G_n[X] \equiv G_n[X/\mu X(\vec{x}).F_n[X./\neg P])],$$

where the arguments in X bound by the fixpoint operator are mapped to the actual variables of the substituted predicates.

Generalised Ackermann's lemma allows us to eliminate existentially quantified variables in cases where there is no first-order formula equivalent to the input, but a formula in fixpoint logics. This approach is followed in the DLS* system.

Whereas Ackermann's lemma plays a central role in both approaches, resolution-based and rewriting-based, we are not aware of any resolution-based system that uses generalised Ackermann's lemma. The methods presented in the thesis can be seen as an exception, since they use both resolution and generalised Ackermann's lemma in order to efficiently compute finite results for all input ontologies in the language under consideration.

2.3 Uniform Interpolation in Modal Logics

Many modal logics are syntactic variants of description logics, which is why results for uniform interpolation in modal logics apply to description logics as well. More precisely, formulae in the multi-modal logic K are syntactic variants of concepts in the description logic $\mathcal{ALC}$, and formulae in hybrid logics correspond to concepts in description logics with nominals. In contrast, formulae in modal logic K can be seen as $\mathcal{ALC}$ concepts with only one role. Reasoning on standard modal logic formulae can therefore be seen as reasoning on description logic concepts in absence of a TBox. Modal logic K is the weakest logic in the class of modal logics. Extensions of modal logic K are obtained by adding additional axioms on the semantics of the modalities.

The modal μ-calculus, which extends modal logic K with fixpoint operators, is expressive enough to represent TBoxes. These can be encoded using the greatest fixpoint operator. The modal μ-calculus is therefore equally expressive as the description logic $\mathcal{ALC}\mu$, which is the extension of the description logic $\mathcal{ALC}$ with fixpoint operators.

In modal logic, forgetting has been studied under the name uniform interpolation. In Craig (1957), William Craig first formulated what is nowadays referred to as the Craig interpolation lemma for first-order logics. Given two first-order logic formulae F_1 and F_2, there is always a first-order logic formula F_i that only uses predicate symbols

that both occur in F_1 and F_2 and for which we have $F_1 \models F_i$ and $F_i \models F_2$. For a logic with this property, we say it has *Craig interpolation*, and we refer to F_i as the Craig interpolant of F_1 and F_2.

Henkin (1963) defines a stronger property, which is commonly known as uniform interpolation, and that can be stated as follows:

Definition 2.3.1. A logic $\mathcal{L}$ has *uniform interpolation* iff for every $\mathcal{L}$ formula F and every set of symbols $\mathcal{S}$ there exists an $\mathcal{L}$ formula $F^{\mathcal{S}}$ such that for every $\mathcal{L}$ formula G that only contains symbols from $\mathcal{S}$ we have $F^{\mathcal{S}} \models G$ iff $F \models G$.

In the above definition, $F^{\mathcal{S}}$ is a *uniform interpolant of F for $\mathcal{S}$*. By comparing this definition to Definition 2.1.2, one can see that uniform interpolants of F for $\mathcal{S}$ are equivalent to results of weakly forgetting from F the symbols that are not in $\mathcal{S}$. Therefore, in logics with uniform interpolation, the result of weakly forgetting any set of predicates is always finite. Henkin proves this property for propositional logic. From the results presented in the last sections it follows that first-order logic does not have uniform interpolation.

First research on uniform interpolation in modal logics focused on determining which logics enjoy the uniform interpolation property. Ghilardi and Zawadowski (1995) prove that the modal logic S4 does not have uniform interpolation. In contrast, Visser (1996) shows that the modal logics K, IPC, GL and S4Grz have uniform interpolation. In D'Agostino and Hollenberg (1996), it is proved that the modal μ-calculus has uniform interpolation as well.

These results are obtained by model-theoretic observations, and leave open the problem of how to construct uniform interpolants for these logics. This question was targeted in Bílková (2007) and Kracht (2007), who describe methods to effectively compute uniform interpolants in the modal logics K and T, in the first case using a sequent-calculus, and the second case using a tableau method. For the modal μ-calculus, a method is presented in D'Agostino and Lenzi (2006) that uses a special form of disjunctive normal form to construct uniform interpolants of formulae in the modal μ-calculus.

All these methods are based on transformations of the input into disjunctive normal form. While this is explicit in the method by D'Agostino and Lenzi (2006), the methods by Bílková (2007) and Kracht (2007) involve an implicit transformation into

disjunctive normal form by using respectively sequent-based and tableau-based reasoning. Computing uniform interpolants using disjunctive normal form has also been used in propositional logic (see Section 2.1). Herzig and Mengin (2008) point out that in a lot of applications, disjunctive normal forms are not a natural way to represent formulae. Instead of using disjunctions, it is often more natural to represent information using conjunctions as the outermost connectives. While Herzig and Mengin are mainly interested in uniform interpolation in the modal logic K, we note this is also, and especially, the case for description logics, since ontologies are usually conjunctions of axioms. Herzig and Mengin present a method that works on modal logic formulae in a conjunctive normal form, using a resolution calculus presented in Enjalbert and Fariñas del Cerro (1989), which we briefly describe in Section 2.7.1. Similar to resolution-based methods for variable elimination in propositional logic, and the SCAN algorithm for second-order quantifier elimination, the method by Herzig and Mengin computes uniform interpolants by resolving on the symbols to be eliminated, which makes it possible to eliminate these symbols in a goal-oriented manner.

2.4 Uniform Interpolation in DL-Lite and $\mathcal{EL}$

The first method for forgetting and uniform interpolation that was explicitly targeted at ontologies is presented in Wang et al. (2008, 2010b). This method targets ontologies and knowledge bases represented in the lightweight description logic DL-Lite. The main restriction of the various dialects of DL-Lite in comparison to other description logics is that it does not allow for concepts under quantifiers. For this reason, forgetting concept and role symbols can effectively be handled in a similar way as for propositional logic. The described method basically uses a resolution algorithm on the symbols to be eliminated to compute the uniform interpolants. Whereas most later work focuses on forgetting in TBoxes, Wang et al. (2008) already consider knowledge bases that also have an ABox.

Konev et al. (2009b) present a uniform interpolation method for the light-weight description logic $\mathcal{EL}$ extended with role hierarchies and domain and range restrictions. In general, $\mathcal{EL}$ does not have uniform interpolation. For this reason, the authors formulate acyclicity conditions on the input that ensure that the computed uniform

interpolant is always finite. The output of this algorithm is in the worst case of exponential size in the size of the input ontology. The authors evaluate their method on two large prominent $\mathcal{EL}$ ontologies, and show that it is able to eliminate large sets of symbols from these. However, the fact that $\mathcal{EL}$ is a Horn logic, and that the method assumes the input to be acyclic, makes the problem much easier than for more expressive description logics and general ontologies.

Lutz et al. (2010) analyse properties of $\mathcal{EL}$ enriched with greatest fixpoint operators, $\mathcal{EL}\nu$, and show that it is as expressive as $\mathcal{EL}$ enriched with simulation quantifiers, a language in which uniform interpolants of concepts can be expressed directly. They further show that Craig interpolants always exist in $\mathcal{EL}\nu$. In Nikitina (2011), an approach for uniform interpolation of general $\mathcal{EL}$ ontologies is presented that also uses greatest fixpoint operators. The method is based on computing most specific superconcepts and most general sub-concepts of the concepts in the signature, which is performed by analysis of derivation graphs for these concepts.

A method for computing uniform interpolants of general $\mathcal{EL}$ TBoxes without using fixpoints is presented in Nikitina and Rudolph (2012, 2014), using a method based on proof-theory and regular tree grammars. The authors prove that these uniform interpolants are in the worst case of triple exponential size in size of the input ontology.

Lutz et al. (2012) additionally target the question of when uniform interpolants of $\mathcal{EL}$ TBoxes can be represented finitely and purely in $\mathcal{EL}$. They develop an algorithm based on tree automata representations of the TBox that can decide whether such a uniform interpolant for a given TBox and signature exists. This way, they prove that the complexity of this problem is ExpTime-complete. If a uniform interpolant exists, it can be computed by giving the set entailed axioms in the selected signature up to a certain depth.

2.5 Uniform Interpolation in $\mathcal{ALC}$

For $\mathcal{ALC}$, uniform interpolation was first described on the level of concepts, in so-called concept interpolants.

Definition 2.5.1. Let C be an $\mathcal{L}$ concept and $\mathcal{S}$ a signature. The concept $C^{\mathcal{S}}$ is an $\mathcal{L}$ *concept uniform interpolant of* C *for* $\mathcal{S}$ if $C^{\mathcal{S}}$ is an $\mathcal{S}$ concept, $\models C \sqsubseteq C^{\mathcal{S}}$ and

$\models C^{\mathcal{S}} \sqsubseteq C'$ for every $\mathcal{S}$ concept C' with $\models C \sqsubseteq C'$.

While for $\mathcal{ALC}$ concepts, a finite uniform interpolant in $\mathcal{ALC}$ always exists, for $\mathcal{ALC}$ TBoxes this is not the case, since $\mathcal{ALC}$ does not have uniform interpolation.

Concept uniform interpolation in $\mathcal{ALC}$ directly corresponds to uniform interpolation in modal logic **K**. Therefore, concept uniform interpolants can be constructed by transforming C into disjunctive normal form and replacing every literal that is not in the signature with $\top$, as it is implicitly done in modal logic K in Bílková (2007) and Kracht (2007). This technique is used in ten Cate et al. (2006) and Wang et al. (2009) for computing concept uniform interpolants of $\mathcal{ALC}$ concepts.

Based on this idea, Wang et al. (2009) develop a method for uniform interpolation of $\mathcal{ALC}$ knowledge bases, which under certain syntactical restrictions can also deal with ABoxes. The method can be compared to the approach by D'Agostino and Lenzi (2006) for computing uniform interpolants in the modal μ-calculus. However, whereas D'Agostino and Lenzi (2006) assume the input to be in a specific disjunctive normal form, Wang et al. (2009) do the transformation incrementally during the computation of the uniform interpolant. The main idea is to represent the TBox as a single concept in disjunctive normal form, so that concepts and roles can easily be eliminated. Since such a representation is possibly infinite, they approximate it incrementally and determine using equivalence tests between increments whether a uniform interpolant has already been computed. The different approximation steps correspond to results of bounded forgetting, as it is defined in Zhou and Zhang (2011) for first-order logic (see Section 2.1).

In Wang et al. (2010a, 2014), this method is optimised using tableau-based reasoning. Classically, a tableau reasoner tries to disprove a formula by exploring different interpretations in different branches until a model is found or every branch contains a contradiction. Wang et al. (2010a) modify this approach for $\mathcal{ALC}$ in such a way that the set of satisfiable branches in the tableau can be transformed back into an approximation of the input TBox which is of a form similar to a disjunctive normal form. By excluding elements from the branches that are not in the desired signature, one can obtain an approximation of the uniform interpolant. Similar to their approach in Wang et al. (2009), the tableau is stepwise extended until two succeeding approximations are equivalent or a maximum number of iterations is reached.

As with the first methods for uniform interpolation in modal logics, a major disadvantage of these two approaches is that they explicitly or implicitly transform the input into disjunctive normal form. This is an unusual representation for TBoxes, which are usually represented as sets of relatively small axioms, rather than one large axiom in the form of a disjunction. The original structure of the ontology is usually not preserved by this approach. Motivated by this, Wang et al. (2010a, 2014) present a second method based on tableaux, which aims at preserving the original structure of the ontology. For this structure-preserving uniform interpolation method, the authors do not provide a termination condition, whereas for the other method it is given in form of an upper bound on the number of required approximation steps.

For the methods that are not structure-preserving, it is claimed in Wang et al. (2009) and in Wang et al. (2010a, 2014) that after 2^n iterations either a uniform interpolant is computed or there is no finite uniform interpolant. From this follows that the size of a uniform interpolant is at most double exponential in the size of the input ontology. This claim is disproved in Lutz and Wolter (2011), where a family of ontologies is identified for which the smallest uniform interpolant is of size in $O(2^{2^{2p(n)}})$, where p is a polynomial and n is the size of the input ontology. Lutz and Wolter (2011) modify the approach in Wang et al. (2009) to compute uniform interpolants of triple exponential size, this way providing a tight bound on the worst case complexity of the size of uniform interpolants. Lutz and Wolter (2011) further prove the complexity of deciding whether a finite uniform interpolant in $\mathcal{ALC}$ exists to be 2ExpTime-complete.

The methods in Wang et al. (2009, 2010a, 2014); Lutz and Wolter (2011) have two drawbacks. First, except for the structure-preserving approach in Wang et al. (2010a, 2014), which does not have a termination criterion, they compute uniform interpolants in disjunctive normal form. As observed earlier, this is an unusual representation, since usually ontologies correspond to large conjunctions of axioms rather than disjunctions. Secondly, they do not follow a goal-oriented approach, which is why these methods are not feasible for large ontologies.

The first approach to overcome these problems in order to obtain practicality is presented in Ludwig and Konev (2013, 2014), which uses resolution to eliminate concept symbols from $\mathcal{ALC}$ ontologies. The method was developed independently around the same time as our method, presented in Chapter 4, was developed, and published two

months earlier than our results (Koopmann and Schmidt, 2013a). It is based on the resolution-based technique by Herzig and Mengin (2008) for computing uniform interpolants of modal logic formulae (see Section 2.3). For this, they extend the resolution calculus by Enjalbert and Fariñas del Cerro (1989), which is also used by Herzig and Mengin, to incorporate TBox axioms. The resulting method works on a conjunctive normal form of the TBox, with the effect that the computed uniform interpolants are of a form that is more natural. This also reduces the normalisation cost compared to a translation into disjunctive normal form. Moreover, using resolution makes it possible to directly compute inferences on the symbols to be eliminated, as it is done for example in the SCAN approach for second-order quantifier elimination (see Section 2.2). The authors show that the method is practical enough to forget small sets of concept symbols from various ontologies, and demonstrate its application for computing logical differences of ontology versions on the NCI thesaurus (for logical difference, see Section 1.1). For ontologies that follow certain acyclicity conditions, the method always terminates. For general ontologies, the method cannot guarantee termination, even if finite uniform interpolants exist. The authors show how the method can be used for bounded forgetting in this case, such that approximated uniform interpolants are computed which preserve all entailments of the uniform interpolant up to a specified depth.

2.6 Related Problems in Description Logics

Besides uniform interpolation, there are various other research topics that are concerned with entailments of ontologies in a specified signature, most notably module extraction, inseparability and logical difference.

Module extraction deals with the problem of extracting subsets of an ontology, so-called *modules*, usually with regard to some selected signature. Logic-based notions of modules require the module to preserve certain entailments in the specified signature, often under specific robustness conditions. Module extraction has applications in ontology reuse, but has also been used to optimise reasoning, such as by the systems Chainsaw (Tsarkov and Palmisano, 2012) and MORe (Romero et al., 2012). Module extraction has received a lot of attention in the last years, as reflected in a volume

on modular ontologies (Stuckenschmidt et al., 2009) and the series of Workshops on Modular Ontologies (WoMO) (Haase et al., 2006; Baclawski et al., 2014).

Logical properties of module extraction have for example been studied in Konev et al. (2008a); Grau et al. (2008). Grau et al. (2008) study a notion of module where a module must be able to replace an ontology in a union of ontologies, such that all entailments over a specified signature in the description logic under consideration are preserved. Notably, deciding modularity according to this notion is undecidable for the description logic $\mathcal{ALCO}$. If modules are required to preserve entailments in second-order logic, deciding modularity is already undecidable for the lightweight description logic $\mathcal{EL}$ (Lutz and Wolter, 2010). Despite these negative results, it is often possible to compute approximations of minimal modules. Methods for module extraction use different methodologies, and for example use syntactical approaches (Grau et al., 2008; Sattler et al., 2009), QBF-solvers (Kontchakov et al., 2010; Gatens et al., 2014) or Datalog reasoning (Romero et al., 2015), and can also involve rewriting of the axioms of the ontology to obtain a more succinct representation of the modules (Nikitina and Glimm, 2012).

Strongly related to the logic-based notions of module extractions is the notion of *inseparability*. Given two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ and a signature $\mathcal{S}$, $\mathcal{O}_1$ and $\mathcal{O}_2$ are $\mathcal{S}$-*inseparable* if they share all entailments over $\mathcal{S}$ in some query language, where different notions of inseparability differ in the query language used. Inseparability can be used to define notions of modules, and has furthermore applications in ontology reuse, ontology refinement and comparison of ontology versions. If two ontologies are not $\mathcal{S}$-inseparable, one can compute finite representations of *the logical difference*, which contains all entailments in $\mathcal{S}$ in which the two ontologies differ. The logical difference can serve as an explanation on why two ontologies are not inseparable, which makes it a powerful tool in analysing ontology changes.

The complexity of different notions of inseparability and the related problem of deciding conservative extensions has been studied for various logics (Lutz and Wolter, 2007; Ghilardi et al., 2006; Lutz et al., 2007; Lutz and Wolter, 2010; Botoeva et al., 2014; Kontchakov et al., 2010). In a lot of description logics, deciding inseparability with respect to entailments in the language under consideration is one exponential harder than deciding satisfiability (Lutz and Wolter, 2007; Ghilardi et al., 2006; Lutz

et al., 2007). For the description logic $\mathcal{ALCOIQ}$, it is undecidable (Lutz et al., 2007). If semantic notions of inseparability based on models are used, inseparability is undecidable already for $\mathcal{EL}$ (Lutz and Wolter, 2010). Methods for computing logical difference have been investigated in Konev et al. (2008b, 2012); Ludwig and Konev (2013, 2014), where notably Ludwig and Konev (2013, 2014) use uniform interpolation for $\mathcal{ALC}$ ontologies to compute logical differences.

2.7 Saturation-Based Reasoning

After practical methods for uniform interpolation, another major contribution of the thesis is the development of new saturation procedures for description logics. Using saturation-based reasoning for generating uniform interpolants has several advantages, of which the following are the most important. (1) Direct methods for resolution work by computing entailments that are represented in the logic under consideration. The computed entailments can therefore be used as they are in the uniform interpolant, which avoids expensive translation procedures. (2) The resolution rule, which usually plays a central role in these approaches, provides a way of computing inferences on a single symbol in a goal-oriented manner, which is advantageous if the aim is to forget that symbol.

Saturation-based reasoning is used by some of the most successful theorem provers for first-order logic, such as E (Schulz, 2002), SPASS (Weidenbach et al., 2002) and VAMPIRE (Riazanov and Voronkov, 2002), and has recently received increased attention for reasoning in description logics.

Since modal logics and description logics are closely related, we briefly discuss the current state of the art of saturation-based reasoning on both modal and description logics.

2.7.1 Modal Logics

Since modal logic formulae can be represented in first-order logic, one obvious approach is to translate a modal logic formula into first-order logic and use an existing saturation-based first-order prover. This approach has been extensively studied in Schmidt (1996, 1999); Hustadt and Schmidt (2002); Schmidt and Hustadt (2003a,b);

Hustadt et al. (2004b); Areces et al. (2001b), and even used as a framework for studying new deduction methods (Schmidt, 2009).

Whereas these approaches translate modality operators using binary predicates, a different representation is chosen in Ohlbach (1988), where a resolution-based method for first-order modal formulae is presented. Ohlbach represents paths along the modalities using additional attributes to the predicates, similar to Skolemisation approaches used in first-order logic theorem proving (see Baaz et al., 2001; Nonnengart and Weidenbach, 2001).

In contrast to translation based approaches, direct resolution works directly on the logic under consideration. Most research on direct resolution methods for modal logics was undertaken throughout the eighties and the early nineties. The first direct resolution method for modal logics is presented in Fariñas del Cerro (1982, 1985); Enjalbert and Fariñas del Cerro (1989). Here, the input formula is expected to be in a CNF-based clausal normal form that allows for nested formulae inside modality operators. In order to be able to resolve on nested clauses, a system of meta-rules is defined, that generates rules which can resolve on arbitrarily nested literals. For this approach, refinements and strategies such as subsumption deletion have been investigated in Auffray et al. (1990), even though completeness of resolution with subsumption deletion was left open in this paper. This problem was solved in the extension of this method used in Ludwig and Konev (2013, 2014) to compute uniform interpolants of $\mathcal{ALC}$ ontologies.

Mints (1989, 1990) avoids the problem of nested formulae by using a flattened normal form, which forbids arbitrary nestings of modal operators. This leads to a simpler calculus without the need for meta-rules, and does not affect the applicability of the calculus, since every set of modal formulae can be translated into an equisatisfiable set of flattened clauses via the introduction of new symbols. A similar idea is followed by the more recent approach for direct resolution for modal logics presented in Nalon and Dixon (2007); Nalon et al. (2014), which applies to a variety of different modal logics.

Whereas all these methods require the input to be normalised in some way, there are also methods that work directly on the input formulae. Melvin Fitting, mostly known for his research on tableau reasoning, presents a direct resolution calculus for

modal logics in Fitting (1990), where transformations on the input formulae are applied during the inference process. This method uses a splitting rule to explore different branches of the modal formulae, which makes it similar to tableau-based reasoning approaches. Areces et al. (2001a) in contrast present a method for modal, description and hybrid logics in negation normal form, which works without a splitting rule, and instead uses labelled formulae to keep track of the different possible branches along the modalities. The authors also generalise their method to the description logic $\mathcal{ALCR}$, which is $\mathcal{ALC}$ extended with conjunctions on roles. In Areces and Gorín (2011), this approach is further refined using ordered resolution with selection functions, closely inspired by the work of Bachmair and Ganzinger (2001).

2.7.2 Description Logics

Even though tableau-based reasoning has long been a leading paradigm in reasoning for description logics, with reasoners such as RACER (Haarslev and Möller, 2003), Pellet (Sirin et al., 2007), and HermiT (Shearer et al., 2008) as prominent examples, in the last years saturation-based reasoning methods received increased attention by the description logic community. First methods using resolution for description logic were based on translations of description logics into first-order logic (Hustadt and Schmidt, 2000; Hustadt et al., 2004b; Schmidt, 2009; Kazakov and Motik, 2008) or into disjunctive datalog (Hustadt et al., 2004a). In recent years, however, researchers started investigating direct saturation methods for description logics, which are often referred to as *consequence based reasoning methods*.

Saturation-based reasoning systems have proved particularly successful for classifying ontologies. Ontology classification is the task of inferring all atomic concept inclusions of the form $A \sqsubseteq B$ that are entailed by an ontology, in order to create a subsumption hierarchy of the concept names in the ontology. Testing these entailments individually is not efficient, since there are quadratically many, and larger ontologies can contain 10,000s of concept names and more. Using saturation-based approaches, it is sufficient to saturate the set of axioms once, until all atomic concept inclusions are derived.

The first consequence-based methods in description logics exploited the simple

structure of the Horn description logic $\mathcal{EL}$ (Baader et al., 2005), most prominently implemented by the ELK reasoner (Kazakov et al., 2012, 2014). The next consequence-based methods extended this approach to the more expressive Horn logics Horn $\mathcal{SHIQ}$ (Kazakov, 2009) and Horn $\mathcal{SROIQ}$ (Ortiz et al., 2010), the full Horn fragment of the web ontology standard language OWL DL 2.0. By using ordered resolution techniques, efficient classification algorithms based on saturation could be developed for the non-Horn logics $\mathcal{ALCH}$ (Simančík et al., 2011a) and $\mathcal{ALCI}$ (Simančík et al., 2011b). The most expressive description logic so far for which a purely consequence-based method has been developed is $\mathcal{SHIQ}$, for which a method is described in Bate et al. (2015). Notably, this method works on first-order representations of the ontology, whereas the other consequence-based approaches work directly on axioms in description logic syntax. By tightly integrating tableau reasoning into consequence-based reasoning, Steigmiller et al. (2014a) develop a saturation-based method for $\mathcal{SROIQ}$, the description logic underlying OWL 2.0 DL, which is implemented by the reasoner Konclude (Steigmiller et al., 2014b). Similar to the methods by Mints (1989, 1990); Nalon and Dixon (2007); Nalon et al. (2014) for direct resolution in modal logics, consequence-based reasoners for description logics usually use a flattened normal form representation of the input, which limits required inferences on nested concepts.

The methods presented in this section often achieve worst-case optimal reasoning time by focusing on the inferences required for the targeted reasoning tasks. They therefore only ensure completeness for the reasoning task in question, and cannot be used to compute all entailments required for uniform interpolation. Herzig and Mengin (2008) as well as Ludwig and Konev (2013, 2014) solve this problem by extending the resolution calculus by Enjalbert and Fariñas del Cerro (1989) with additional rules, but for $\mathcal{ALC}$, the resulting calculus does not terminate in case the uniform interpolant cannot be finitely represented in $\mathcal{ALC}$. For these reasons, we introduce a new family of saturation procedures in this thesis that overcomes these problems.

Chapter 3

Basics of Description Logics and Uniform Interpolation

We assume basic knowledge of first-order logic and set theory. Given a set or multiset N, we denote by $\#N$ the number of elements in N. An *ordering* is any binary relation that is irreflexive and transitive. An ordering $\prec$ is *total* iff for any two distinct elements a, b on which the ordering is defined, we either have $a \prec b$ or $b \prec a$. If $\prec$ is an ordering on elements in a set N, we denote by $\prec_{mul}$ the *multiset extension of* $\prec$, which is defined as follows. For two multisets or sets $N_1, N_2 \subseteq N$, if there exists an element $x_2 \in (N_2 \setminus N_1)$ such that $x_1 \prec x_2$ for all $x_1 \in (N_1 \setminus N_2)$, then $N_1 \prec_{mul} N_2$. Given a total ordering $\prec$ on elements of a set N, we define the *n-ary lexicographic extension* $\prec^n$ on the n-ary product $N \times \ldots \times N$ as the ordering that satisfies $\langle x_1^a, \ldots, x_n^a \rangle \prec^n \langle x_1^b, \ldots, x_n^b \rangle$ iff there is an index m such that $x_i^a = x_i^b$ for all $i < m$ and $x_m^a \prec x_m^b$.

For any expression E, $E[F_1/F_2]$ is the result of replacing every subexpression F_1 in E by F_2. If $E[F]$ is an expression which contains a subexpression F, $E[F_1]$ denotes the result of replacing F in $E[F]$ by F_1.

3.1 Description Logics

We introduce the description logics discussed throughout the thesis. An overview of basic and more expressive description logics can be found in Baader and Nutt (2007) and Calvanese and Giacomo (2007). Description logics can be viewed as fragments of first-order logic that are used to represent ontologies. Description logics represent

information by means of *concepts*, *roles* and *individuals*. There are various description logics that differ in expressivity as well as in reasoning complexity. We first give a general overview of the syntax and semantics of all classical description logic operators that play a role in the thesis. In Section 3.3 we introduce fixpoint operators, which are less common, but play a central role in our representation of uniform interpolants.

We first define the description logic $\mathcal{SHOIQ}$, which is the union of all classical description logics considered. We start by introducing the operators that are used to express concepts and roles. We define N_c, N_r and N_i to be three pair-wise disjoint sets of *concept symbols*, *role symbols* and *individual symbols*. A role is either a role symbol or is of the form r^-, where $r \in N_r$. If R is a role, we define the *inverse* $\mathsf{Inv}(R)$ of R by $\mathsf{Inv}(r) = r^-$ and $\mathsf{Inv}(r^-) = r$, for all $r \in N_r$.

A *concept* is an expression of one of the following forms, where $A \in N_c$, C, C_1 and C_2 are any concepts, R is any role, n and m are natural numbers with $n \geq 1$, $m \geq 0$, and $a_1, \ldots, a_n \in N_i$.

$$\top \mid \bot \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2$$
$$\exists R.C \mid \forall R.C \mid \geq nR.C \mid \leq nR.C$$
$$\{a_1, \ldots, a_n\}$$

The concepts in the second row are called *role restrictions* and referred to from left to right as *existential restrictions*, *universal restrictions*, *$\geq$-number restrictions* and *$\leq$-number restrictions*. We alternatively refer to number restrictions as *cardinality restrictions*. A concept C occurs *positively in* C' iff C occurs in C' only under an even number of negations. If C occurs in C' only under an odd number of negations, it occurs *negatively in* C'.

In the remainder of the thesis, we use the symbols A and B to denote concept symbols, r and s to denote role symbols, a and b to denote individuals, C to denote concepts and R and S to denote roles, in each case possibly using subscripts or prime symbols.

An *RBox* $\mathcal{R}$ defines relations between roles using *role inclusion axioms*, *role equivalence axioms* and *transitivity axioms*, which respectively have the following forms.

$$R \sqsubseteq S$$
$$R \equiv S$$
$$\mathsf{trans}(R),$$

where R and S are any roles. $R \equiv S$ is a short hand of the two role inclusion axioms $R \sqsubseteq S$ and $S \sqsubseteq R$. Given an RBox $\mathcal{R}$, we denote by $\sqsubseteq_{\mathcal{R}}$ the smallest reflexive-transitive relation such that $R \sqsubseteq_{\mathcal{R}} S$ and $\mathsf{Inv}(R) \sqsubseteq_{\mathcal{R}} \mathsf{Inv}(S)$ for all $R \sqsubseteq S \in \mathcal{R}$. In an RBox $\mathcal{R}$ with $R \sqsubseteq_{\mathcal{R}} S$, we call R a *sub-role* of S and S a *super-role* of R. A role R is *transitive* in $\mathcal{R}$ if $\mathsf{trans}(R) \in \mathcal{R}$ or $\mathsf{trans}(\mathsf{Inv}(R)) \in \mathcal{R}$. A role is *simple* if it does not have a transitive sub-role. Otherwise, it is *complex*.

A *TBox* $\mathcal{T}$ contains terminological knowledge in form of a set of *concept inclusion axioms* and *concept equivalence axioms* of the following respective forms.

$$C_1 \sqsubseteq C_2$$
$$C_1 \equiv C_2,$$

where C_1 and C_2 are any concepts. An *ABox* contains assertional knowledge about individuals in form of a set of *concept assertions* and *role assertions* of the following respective forms.

$$C(a)$$
$$R(a, b),$$

where C is a concept, R a role and $a, b \in N_i$. The elements of TBoxes, RBoxes and ABoxes are collectively referred to as *axioms*.

An *ontology* $\mathcal{O}$ is a tuple $\langle \mathcal{T}, \mathcal{R} \rangle$, where $\mathcal{T}$ is a TBox and $\mathcal{R}$ is an RBox. A *knowledge base* $\mathcal{K}$ is a tuple $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, where $\mathcal{T}$ is a TBox, $\mathcal{R}$ is an RBox and $\mathcal{A}$ is an ABox. In order to ensure decidability of the logic, we further require that only simple roles occur in number restrictions in the TBox of an ontology or knowledge base (see Horrocks et al., 2000). In other words, if $\mathcal{O} = \langle \mathcal{T}, \mathcal{R} \rangle$ or $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, for any concept of the form $\geq nR.C$ or $\leq nR.C$ occurring in $\mathcal{T}$, there is no role S with $S \sqsubseteq_{\mathcal{R}} R$ and $\mathsf{trans}(S) \in \mathcal{R}$ or $\mathsf{trans}(\mathsf{Inv}(S)) \in \mathcal{R}$. If $\mathcal{R}$ is the RBox of an ontology $\mathcal{O}$

Basic Description Logics	
Name	Allowed Constructs
$\mathcal{EL}$	A, $C \sqcap D$, $\exists R.C$, $\top$
$\mathcal{ALC}$	$\mathcal{EL}$ + $\neg C$
$\mathcal{S}$	$\mathcal{ALC}$ + $trans(R)$

Extensions		
Name	Allowed Constructs	Description
$\mathcal{H}$	$R \sqsubseteq S$, $R \equiv S$	Role Hierarchies
$\mathcal{O}$	$\{a_0, ..., a_n\}$	Nominals
$\mathcal{I}$	r^-	Inverse Roles
$\mathcal{F}$	$\leq 1 R.\top$	Functional role restrictions
$\mathcal{Q}$	$\geq n R.C$, $\leq n R.C$	Qualified Number Restrictions

Table 3.1: Description logic families and their extensions.

or a knowledge base $\mathcal{K}$, we alternatively denote the reflexive-transitive closure $\sqsubseteq_{\mathcal{R}}$ of the role inclusion axioms in $\mathcal{R}$ by $\sqsubseteq_{\mathcal{O}}$ and $\sqsubseteq_{\mathcal{K}}$, respectively.

By restricting which operators can be used for constructing concepts and roles, we obtain different description logics. Table 3.1 gives an overview of the operators allowed in the description logics $\mathcal{EL}$, $\mathcal{ALC}$ and $\mathcal{S}$, as well as additional operators allowed in the extensions of these. A description logic is usually denoted by adding postfixes that describe the additional allowed operators to one of these description logics, such as in $\mathcal{ELI}$ for $\mathcal{EL}$ with inverse roles, $\mathcal{ALCF}$ for $\mathcal{ALC}$ with functional role restrictions, and $\mathcal{SHQ}$ for $\mathcal{S}$ with role hierarchies and number restrictions. Note that role inclusion axioms are only allowed in description logics with role hierarchies (denoted by the postfix $\mathcal{H}$), and transitivity axioms are only allowed in $\mathcal{S}$ and its extensions. Let $\mathcal{L}$ be any description logic. We speak of an $\mathcal{L}$ ontology, an $\mathcal{L}$ axiom and a $\mathcal{L}$ concept to denote that the ontology, axiom or concept only uses operators that are allowed in $\mathcal{L}$.

The semantics of description logics is defined using Tarski-style interpretations. An *interpretation* is a tuple $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where the *domain* $\Delta^{\mathcal{I}}$ is a non-empty set of *domain elements*, and the *interpretation function* $\cdot^{\mathcal{I}}$ is a function that maps concept symbols $A \in N_c$ to subsets $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ of the domain, role symbols $r \in N_r$ to binary relations $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ over the domain, and individuals $a \in N_i$ to elements $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ of the domain. $\cdot^{\mathcal{I}}$ is extended to concepts and roles according to Table 3.2.

An axiom α is *true* in an interpretation $\mathcal{I}$, in symbols $\mathcal{I} \models \alpha$, iff one of the following conditions hold:

Description Logic Expression E	Interpretation $E^{\mathcal{I}}$
$\bot$	$\emptyset$
$\top$	$\Delta^{\mathcal{I}}$
$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
$\exists R.C$	$\{x \mid \exists y : (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
$\forall R.C$	$\{x \mid \forall y : (x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
$\geq n R.C$	$\{x \mid \#\{(x,y) \in R^{\mathcal{I}} \mid y \in C^{\mathcal{I}}\} \geq n\}$
$\leq n R.C$	$\{x \mid \#\{(x,y) \in R^{\mathcal{I}} \mid y \in C^{\mathcal{I}}\} \leq n\}$
$\{a_1, \ldots, a_n\}$	$\{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\}$
r^-	$\{(x,y) \mid (y,x) \in r^{\mathcal{I}}\}$

Table 3.2: Interpretation of roles and concepts.

1. α is of the form $C_1 \sqsubseteq C_2$ and $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$.

2. α is of the form $C_1 \equiv C_2$ and $C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$.

3. α is of the form $R_1 \sqsubseteq R_2$ and $R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$.

4. α is of the form $R_1 \equiv R_2$ and $R_1^{\mathcal{I}} = R_2^{\mathcal{I}}$.

5. α is of the form $\mathsf{trans}(R)$ and $R^{\mathcal{I}}$ is transitive.

6. α is of the form $C(a)$ and $a^{\mathcal{I}} \in C^{\mathcal{I}}$.

7. α is of the form $R(a,b)$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$.

An interpretation $\mathcal{I}$ is a *model* of an ontology $\mathcal{O}$ iff every axiom $\alpha \in \mathcal{O}$ is true in $\mathcal{I}$. If an ontology has a model, it is satisfiable, otherwise it is unsatisfiable. If an axiom α is true in every model of an ontology $\mathcal{O}$, we say α is *entailed* by $\mathcal{O}$, in symbols $\mathcal{O} \models \alpha$. If an axiom is entailed by every ontology $\mathcal{O}$, we say α is *valid*, in symbols $\models \alpha$. If every axiom of an ontology $\mathcal{O}_1$ is entailed by an ontology $\mathcal{O}_2$, we say $\mathcal{O}_1$ is entailed by $\mathcal{O}_2$, in symbols $\mathcal{O}_2 \models \mathcal{O}_1$. Note that we have $\mathcal{O} \models R \sqsubseteq S$ iff $R \sqsubseteq_{\mathcal{O}} S$.

Another way of defining the semantics of description logics is by giving a translation function $\cdot^f$ from ontology axioms to first-order logic formulae, where concept and role symbols are interpreted as predicates and individuals as constants. This function is shown in Table 3.3. Concepts and roles are mapped respectively to formulae with one or two free variables. For an expression E and its translation E^f, $E^f(\vec{t})$ denotes the

Description Logic Expression	FOL translation C^f
$\top$	**true**
$\bot$	**false**
A	$A(x)$
$\neg C$	$\neg C^f(x)$
$\exists R.C$	$\exists y : \big(R^f(x,y) \wedge C^f(y)\big)$
$\forall R.C$	$\forall y : \big(R^f(x,y) \rightarrow C^f(y)\big)$
$C_1 \sqcap C_2$	$C_1^f(x) \wedge C_2^f(x)$
$C_1 \sqcup C_2$	$C_1^f(x) \vee C_2^f(x)$
$\geq n R.C$	$\exists x_1, \ldots, x_n \;:\; \big(\bigwedge_{1 \leq i \leq n} \big(R^f(x,x_i) \wedge C^f(x_i) \wedge \bigwedge_{1 \leq j < i} x_j \neq x_i\big)\big)$
$\leq n R.C$	$\forall x_1, \ldots, x_n \;:\; \big(\big(\bigwedge_{1 \leq i \leq n+1} R^f(x,x_i) \wedge C^f(x_i)\big) \rightarrow \bigvee_{1 \leq i < j \leq n+1} x_i = x_j\big)$
$\{a_1, \ldots, a_n\}$	$x = a_1 \vee \ldots \vee x = a_n$
r	$r(x,y)$
r^-	$r(y,x)$
$C_1 \sqsubseteq C_2$	$\forall x (C_1^f(x) \rightarrow C_2^f(x))$
$C_1 \equiv C_2$	$\forall x (C_1^f(x) \leftrightarrow C_2^f(x))$
$R_1 \sqsubseteq R_2$	$\forall x, y (R_1^f(x,y) \rightarrow R_2^f(x,y))$
$R_1 \equiv R_2$	$\forall x, y (R_1^f(x,y) \leftrightarrow R_2^f(x,y))$
$trans(R)$	$\forall x, y, z ((R^f(x,y) \wedge R^f(y,z)) \rightarrow R^f(x,z))$
$C(a)$	$C^f(a)$
$R(a,b)$	$R^f(a,b)$

Table 3.3: Translation from description logic expressions to first-order logic.

result of mapping the free variables in E to $\vec{t}$, where the first term in $\vec{t}$ is mapped to x and the second to y. Given an ontology $\mathcal{O}$, $\mathcal{O}^f$ refers to the theory obtained by replacing every axiom α by α^f. $\mathcal{O}$ is satisfiable iff $\mathcal{O}^f$ is satisfiable, and $\mathcal{O} \models \alpha$, where α is any axiom, iff $\mathcal{O}^f \models \alpha^f$.

3.2 Uniform Interpolation

A *signature* $\mathcal{S} \subseteq N_c \cup N_r$ is any set of concept and role symbols. The signature $sig(E)$ denotes the set of concept and role symbols occurring in E, where E ranges over concepts, axioms, TBoxes, RBoxes, ABoxes, ontologies and knowledge bases. If for a concept C and a signature $\mathcal{S}$ we have $sig(C) \subseteq \mathcal{S}$, we call C an $\mathcal{S}$ *concept*. In the same way we define the notions $\mathcal{S}$ axiom, $\mathcal{S}$ TBox, $\mathcal{S}$ ontology and $\mathcal{S}$ knowledge base.

We now reformulate the definition of uniform interpolation given in Definition 2.3.1:

Definition 3.2.1. Let $\mathcal{L}$ be a description logic, $\mathcal{K}$ be a knowledge base and $\mathcal{S}$ be a signature. $\mathcal{K}^{\mathcal{S}}$ is an $\mathcal{L}$ *uniform interpolant of* $\mathcal{K}$ *for* $\mathcal{S}$ iff the following conditions hold:

1. $sig(\mathcal{K}^{\mathcal{S}}) \subseteq \mathcal{S}$.

2. For every $\mathcal{L}$ axiom α with $sig(\alpha) \subseteq \mathcal{S}$, we have $\mathcal{K}^{\mathcal{S}} \models \alpha$ iff $\mathcal{K} \models \alpha$.

Uniform interpolants of TBoxes and knowledge bases are defined accordingly.

It is worth noting that this notion of uniform interpolation differs from other definitions found in the literature that usually only require TBox $\mathcal{L}$ axioms in the signature to be preserved by the uniform interpolant (compare Lutz and Wolter, 2011; Ludwig and Konev, 2013; Nikitina and Rudolph, 2014), whereas we require any $\mathcal{L}$ axiom in the signature to be preserved by the uniform interpolant. This does not make a difference in practice, since the mentioned publications only consider uniform interpolants of TBoxes, but not of knowledge bases with RBoxes or ABoxes. A TBox $\mathcal{T}$ on its own only entails RBox axioms of the forms $R \sqsubseteq R$, $R \equiv R$, and ABox axioms $C(a)$ for which $\mathcal{T} \models \top \sqsubseteq C$. These entailments are either tautological or fully determined by the TBox axioms entailed by $\mathcal{T}$. Therefore, while more general, our notion of uniform interpolants is compatible to the notions used in the literature on uniform interpolation of TBoxes.

3.3 Fixpoints

Let N_v be a set of *concept variables* that is disjoint with N_c, N_r and N_i. In description logics with fixpoints, we additionally have concepts of the form X, $X \in N_v$ (concept variable), $\nu X.C[X]$ (least fixpoint) and $\mu X.C[X]$ (greatest fixpoint), where $C[X]$ is a concept in which X occurs only positively, that is, under an even number of negations. In concepts of the forms $\nu X.C[X]$ and $\mu X.C[X]$, we say that the variable X is *bound*. If every variable in a concept is bound, the concept is *closed*, otherwise it is *open*. For all axioms of the form $C_1 \sqsubseteq C_2$, $C_1 \equiv C_2$ and $C(a)$ that occur in a TBox or an ABox, we require the concepts C_1, C_2 and C to be closed.

A fixpoint of a function f is a value x for which $f(x) = x$. Intuitively, in the context of concepts, we understand open concepts of the form $C[X]$ as functions that take as argument a concept C_2 and return a concept $C[C_2]$ that is the result of replacing every occurrence of X in $C[X]$ by C_2. If $C[X]$ contains X only positively, $C[X]$ behaves like a monotone function with respect to the concept inclusion relation, that is, $\mathcal{O} \models C_1 \sqsubseteq C_2$ implies $\mathcal{O} \models C[C_1] \sqsubseteq C[C_2]$ for all concepts C_1 and C_2 and all ontologies $\mathcal{O}$. Furthermore, the relation $\sqsubseteq$ has both a minimal and maximal element, which are the concepts $\bot$ and $\top$, respectively. These two properties, monotonicity of $C[X]$ and the existence of both a minimal and a maximal element, ensure, due to the Knaster & Tarski fixpoint theorem, that $C[X]$ always has a least and a greatest fixpoint, even though it might not always be finitely representable without fixpoint operators. $\mu X.C[X]$ denotes the smallest concept C_μ with respect to $\sqsubseteq$ for which $C[C_\mu] \equiv C_\mu$, and $\nu X.C[X]$ denotes the largest concept C_ν with respect to $\sqsubseteq$ for which $C[C_\nu] \equiv C_\nu$. For example, $\mu X.\exists r.X$ is equivalent to $\bot$, and $\nu X.\forall r.X$ is equivalent to $\top$. $\nu X.\exists r.X$, cannot be equivalently expressed by a finite concept, since we have $\exists r.\top \sqsubseteq \top$, $\exists r.\exists r.\top \sqsubseteq \exists r.\top$, $\exists r.\exists r.\exists r.\top \sqsubseteq \exists r.\exists r.\top$ and so on, which means $\nu X.\exists r.X$ is only equivalent to an infinite concept of the form $\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r. \ldots$.

We define the semantics of fixpoint expressions formally, following Calvanese et al. (1999), and using the Knaster & Tarksi fixpoint theorem. Open concepts are interpreted using a *valuation function* $\rho : N_v \mapsto 2^{\Delta^\mathcal{I}}$ from concept variables to subsets of the domain. Given a valuation function ρ and a set $\mathcal{E} \subseteq \Delta^\mathcal{I}$, $\rho[X/\mathcal{E}]$ is a valuation function identical to ρ except that $\rho[X/\mathcal{E}](X) = \mathcal{E}$. Based on an interpretation $\mathcal{I}$

and a valuation function ρ, we define the *extension function* $\cdot^{\mathcal{I}}_{\rho}$, which maps open concepts to subsets of $\Delta^{\mathcal{I}}$ in the same way as $\cdot^{\mathcal{I}}$, only that concept variables are interpreted according to the valuation function ρ. Furthermore, $\cdot^{\mathcal{I}}_{\rho}$ is defined for fixpoints as follows:

$$(\mu X.C)^{\mathcal{I}}_{\rho} = \bigcap \{\mathcal{E} \subseteq \Delta^{\mathcal{I}} \mid C^{\mathcal{I}}_{\rho[X/\mathcal{E}]} \subseteq \mathcal{E}\}$$
$$(\nu X.C)^{\mathcal{I}}_{\rho} = \bigcup \{\mathcal{E} \subseteq \Delta^{\mathcal{I}} \mid \rho \subseteq C^{\mathcal{I}}_{\rho[X/\mathcal{E}]}\}$$

Note that if C is a closed concept, the valuation function ρ has no influence on $C^{\mathcal{I}}_{\rho}$. We can therefore extend the interpretation function defined in the last section to incorporate fixpoints by setting $C^{\mathcal{I}} = C^{\mathcal{I}}_{\rho}$ for closed concepts C.

For any description logic $\mathcal{L}$ defined in the last section, we denote by $\mathcal{L}\mu$ its extension with least and greatest fixpoint expressions, and by $\mathcal{L}\nu$ its extension with greatest fixpoint expressions, which are only allowed to occur positively in the ontology.

Chapter 4

Practical Uniform Interpolation for $\mathcal{ALC}$

In this chapter, we introduce a new method for uniform interpolation of $\mathcal{ALC}$ ontologies, which overcomes the main problems of the existing methods for $\mathcal{ALC}$ presented in Chapter 2. This method incorporates the central ideas for computing finite representations of uniform interpolants in a goal-oriented manner, and forms the basis of all uniform interpolation methods presented in later chapters. In this chapter, we introduce central notions and definitions that are used to characterise calculi and methods we present in later chapters, most notably the notions of "interpolation completeness" and "uniform interpolation modulo direct cycles". Furthermore, the proofs for soundness, refutational completeness and interpolation completeness of the developed calculi form the basis for corresponding proofs for other calculi introduced throughout the thesis.

In order to infer the axioms that have to be included in the uniform interpolant, we develop a new reasoning calculus. Based on observations in Section 2.5, we find that the following features are desirable for a reasoning method, if we want to use it to compute uniform interpolants in a practical manner.

1. It should be able to infer all axioms that have to be included in the uniform interpolant.

2. It should not infer too many axioms that do not have to be included in the uniform interpolant.

3. It should represent axioms in a representation that can easily be translated into a typical representation in the language under consideration.

4. It should be able to capture infinite entailments, which might have to be preserved in a uniform interpolant, in a finite way.

Feature 1 is a necessary condition for any method that is used to compute uniform interpolants, and is captured in the notion of *interpolation completeness*. This notion is defined formally in Section 4.4. Most reasoning methods are tailored towards specific reasoning problems, such as deciding consistency or computing classification trees, and do not satisfy this property. Moreover, methods for forgetting in first-order logic may omit inferences that are necessary to represent the uniform interpolant in the description logic under consideration.

From a theoretical perspective, Feature 2 is not required for the computation of uniform interpolants. Due to the high worst-case complexity of uniform interpolation, it has however to be fulfilled by any method that is supposed to compute uniform interpolants practically. We are interested in developing practical methods, and therefore it is necessary that the calculus allows us to derive inferences in a goal-oriented manner. A minimal subset of inferences that have to be included in a uniform interpolant is defined formally in Section 4.4. A more implementation-oriented strategy for computing uniform interpolants is given in Chapter 8, where we discuss our implementations of the methods presented in the thesis.

Features 3 and 4 are concerned with representation of the inferred axioms. We want to avoid overhead caused by translations of the input and the output of the method into suitable representations. Moreover, we have to represent any current results in a finite way throughout the computation, even if the uniform interpolant is not finitely representable in the input language. Since a finite representation of the uniform interpolant is not always possible in $\mathcal{ALC}$, we need to represent uniform interpolants differently if we want to obtain finite results on all inputs. Note that in Feature 3, we are not only interested in an easy translation into the language under consideration, but we also want to represent the result in a form that is typical for ontologies. Ontologies are usually represented as conjunctions of relatively small axioms, and optimally, our uniform interpolants should have a similar form. Methods like the ones by Wang et al.

(2014), which compute uniform interpolants in a disjunctive normal form, do not fulfil this criterion. In Section 4.1, we introduce a normal form representation that allows for both, easy translatability and finite representations.

The calculus and interpolation procedure presented in this chapter apply not only to $\mathcal{ALC}$ ontologies, but to $\mathcal{ALC}\nu$ ontologies as well. $\mathcal{ALC}\nu$ has the uniform interpolation property, that is, for every $\mathcal{ALC}\nu$ ontology $\mathcal{O}$ and every signature $\mathcal{S}$, there is a finite $\mathcal{ALC}\nu$ ontology that is the uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$. Since $\mathcal{ALC}\nu$ is a strict super-language of $\mathcal{ALC}$, this also implies that for every $\mathcal{ALC}$ ontology, we can always compute a finite uniform interpolant in $\mathcal{ALC}\nu$.

$\mathcal{ALC}\nu$ ontologies can be translated into normal form and back using simple rewriting rules. Furthermore, for a fixed input ontology, the representation gives us a finite upper bound on the axioms that can be derived using our method, without leaving out required inferences. This ensures termination of the method, as well as a finite representation of the current result at any stage of the computation. The upper bound on axioms that can be inferred is achieved using a bounded dynamic introduction of new symbols during the computation of the uniform interpolant.

For applications that require the uniform interpolant to be represented in a language without fixpoints, we describe two techniques for how uniform interpolants with fixpoints can be represented in pure $\mathcal{ALC}$. The first representation makes use of auxiliary concepts that represent direct cyclical patterns, and is formally captured in the notion of "uniform interpolant modulo direct cycles". The second representation is based on depth-restricted approximation, as it is captured by the definition of bounded forgetting by Zhou and Zhang (2011) (see Section 2.1).

4.1 The Normal Form

4.1.1 Normal Form Transformation

We present the normal form used by our method. The normal form is based on structural transformation and flattening techniques. For this, we define a set $N_d \subset N_c$ of *definer symbols*, *definers* for short, which play a special role in the method. Throughout the paper, we denote definers by the letter D with possible subscripts or prime symbols. Formally, we define ontologies in $\mathcal{ALC}$ normal form as follows.

$$C_1 \sqsubseteq C_2 \quad \Longrightarrow \top \sqsubseteq \neg C_1 \sqcup C_2 \tag{4.1}$$
$$C_1 \equiv C_2 \quad \Longrightarrow \top \sqsubseteq \neg C_1 \sqcup C_2, \quad \top \sqsubseteq \neg C_2 \sqcup C_1 \tag{4.2}$$

$$\top \sqsubseteq C_1 \sqcup \neg\neg C_2 \quad \Longrightarrow \top \sqsubseteq C_1 \sqcup C_2 \tag{4.3}$$
$$\top \sqsubseteq C_1 \sqcup \neg(C_2 \sqcap C_3) \quad \Longrightarrow \top \sqsubseteq C_1 \sqcup \neg C_2 \sqcup \neg C_3 \tag{4.4}$$
$$\top \sqsubseteq C_1 \sqcup \neg(C_2 \sqcup C_3) \quad \Longrightarrow \top \sqsubseteq C_1 \sqcup (\neg C_2 \sqcap \neg C_3) \tag{4.5}$$
$$\top \sqsubseteq C_1 \sqcup (C_2 \sqcap C_3) \quad \Longrightarrow \top \sqsubseteq C_1 \sqcup C_2, \quad \top \sqsubseteq C_1 \sqcup C_3 \tag{4.6}$$
$$\top \sqsubseteq C_1 \sqcup \neg(\exists r.C_2) \quad \Longrightarrow \top \sqsubseteq C_1 \sqcup \forall r.\neg C_2 \tag{4.7}$$
$$\top \sqsubseteq C_1 \sqcup \neg(\forall r.C_2) \quad \Longrightarrow \top \sqsubseteq C_1 \sqcup \exists r.\neg C_2 \tag{4.8}$$

$$\top \sqsubseteq C_1 \sqcup \exists r.C_2 \quad \Longrightarrow \top \sqsubseteq C_1 \sqcup \exists r.D, \quad D \sqsubseteq C_2 \tag{4.9}$$
$$\top \sqsubseteq C_1 \sqcup \forall r.C_2 \quad \Longrightarrow \top \sqsubseteq C_1 \sqcup \forall r.D, \quad D \sqsubseteq C_2 \tag{4.10}$$
$$\top \sqsubseteq C_1 \sqcup \nu X.C_2 \quad \Longrightarrow \top \sqsubseteq C_1 \sqcup A, \quad A \sqsubseteq C_2[X/A] \tag{4.11}$$

Figure 4.1: Transformation rules transforming any $\mathcal{ALC}\nu$ ontology into $\mathcal{ALC}$ normal form.

Definition 4.1.1. An $\mathcal{ALC}$ *literal* is a concept of one of the following forms:

$$A \mid \neg A \mid \exists r.D \mid \forall r.D,$$

where $A \in N_c$, $r \in N_r$ and $D \in N_d$. A literal of the form D, $D \in N_d$, is called *positive definer literal*. A literal of the form $\neg D$, $D \in N_d$, is called *negative definer literal*.

An $\mathcal{ALC}$ *clause* is a concept inclusion of the following form:

$$\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n,$$

where every L_i, $1 \leq i \leq n$, is a literal. We usually omit the leading '$\top \sqsubseteq$' and assume that $\mathcal{ALC}$ clauses are represented as sets, that is, they do not contain duplicate elements and the order of the literals is not important. The empty clause is denoted by $\bot$ and represents a direct contradiction. An ontology is in $\mathcal{ALC}$ *normal form* if every axiom is an $\mathcal{ALC}$ clause.

We denote clauses using the letter C and clause sets using the letters $\mathcal{N}$ and $\mathcal{M}$, possibly using subscripts and prime symbols.

Any $\mathcal{ALC}\nu$ ontology can be transformed into $\mathcal{ALC}$ normal form using the transformations in Figure 4.1, where C_1, C_2 and C_3 are arbitrary, possibly empty, $\mathcal{ALC}\nu$

concepts, and $D \in N_d$ is a fresh definer symbol. The rules are applied modulo commutativity and associativity of $\sqcup$ and $\sqcap$.

The transformations (4.1)–(4.8) are standard CNF transformations. The next two transformations are structural transformations, which introduce one fresh definer symbol D for every concept that is nested under a role restriction. The last transformation is inspired by generalised Ackermann's Lemma (see Section 2.2), but applied in the other direction, that is, introducing a symbol instead of eliminating it. Given an $\mathcal{ALCL}\nu$ ontology $\mathcal{O}$, we denote the result of these transformations by $Cl(\mathcal{O})$. The following theorem follows from known results in normal form transformations, as well as from generalised Ackermann's Lemma.

Theorem 4.1.2. *Given any $\mathcal{ALCL}\nu$ ontology $\mathcal{O}$, any sequence of applications of the rules in Figure 4.1 terminates and computes an ontology $\mathcal{N} = Cl(\mathcal{O})$. $\mathcal{N}$ is in $\mathcal{ALC}$ normal form, and for every axiom α with $sig(\alpha) \subseteq sig(\mathcal{O})$, $\mathcal{N} \models \alpha$ iff $\mathcal{O} \models \alpha$.*

Proof. Transformation rules (4.1)–(4.10) are standard structural transformation rules. The last rule, which eliminates greatest fixpoint expressions, is less common. We note that the transformation looks as follows when represented in first order logic:

$$\forall x\big(C_1(x) \vee \nu X(x').C_2(x)\big) \;\Rightarrow \forall x\big(C_1(x) \vee A(x)\big) \wedge \forall x\big(\neg A(x) \vee C_2(x)[X/A]\big)$$

We observe the similarity to the first equivalence stated in generalised Ackermann's Lemma, where corresponding subformulae occur in the opposite order (see Section 2.2):

$$\exists X.\forall \vec{x}\big(\neg X(\vec{x}) \vee F_p[X]\big) \wedge G_p[X] \equiv G_p[X/\nu X(\vec{x}).F_p[X]]$$

The fixpoint expression on the right in generalised Ackermann's Lemma corresponds to the fixpoint expression on the left in our transformation. It is now easy to see that the transformation preserves all entailments of the original ontology that are not using the concept symbol A. □

Example 4.1.3. Consider the following ontology $\mathcal{O}_1$:

$$A \sqsubseteq B \sqcup C$$

$$B \sqsubseteq \exists r.B$$

$$C \sqsubseteq \forall r.\neg B.$$

The obtained clause set $Cl(\mathcal{O}_1)$ is the following:

1. $\neg A \sqcup B \sqcup C$
2. $\neg B \sqcup \exists r.D_1$
3. $\neg D_1 \sqcup B$
4. $\neg C \sqcup \forall r.D_2$
5. $\neg D_2 \sqcup \neg B$,

where D_1 and D_2 are definers introduced during the structural transformation.

4.1.2 Eliminating Introduced Definer Symbols

Our uniform interpolation method computes entailments in form of $\mathcal{ALC}$ clauses. However, the uniform interpolant itself should not contain definer symbols, since these are not part of the desired signature. For this reason, we introduce a method for eliminating definer symbols that transforms clause sets back into ontologies without definer symbols. First, we define a sufficient condition on clause sets that makes this transformation possible.

Definition 4.1.4. A clause is *normal* if it contains at most one negative definer literal. A set of clauses is normal if every clause in it is normal. Given a set of clauses $\mathcal{N}$, we use the notation $\mathcal{N}_n$ for the set of normal clauses in $\mathcal{N}$.

The clauses generated by the transformation in Section 4.1.1 are all normal, given that the input ontology does not contain definer symbols.

For any set of normal clauses $\mathcal{N}_n$, we can eliminate the definers in it using the rewrite rules shown in Figure 4.2. Observe that the cyclic definer elimination rule introduces greatest fixpoint expressions to the ontology.

Given any set $\mathcal{N}_n$ of normal clauses, we denote the result of these transformations by $Ont(\mathcal{N}_n)$.

Theorem 4.1.5. *Given any set $\mathcal{N}_n$ of normal $\mathcal{ALC}$ clauses, $\mathcal{O} = Ont(\mathcal{N}_n)$ is an $\mathcal{ALC}\nu$ ontology without definer symbols. Moreover, for every axiom α with $sig(\alpha) \cap N_d = \emptyset$, $\mathcal{O} \models \alpha$ iff $\mathcal{N}_n \models \alpha$.*

Declausification:

$$\frac{\mathcal{O} \cup \{\neg D \sqcup C_1, \ldots, \neg D \sqcup C_n\}}{\mathcal{O} \cup \{D \sqsubseteq C_1 \sqcap \ldots \sqcap C_n\}} \quad \text{provided } D \text{ occurs only positively in } \mathcal{O}$$

Non-cyclic definer elimination:

$$\frac{\mathcal{O} \cup \{D \sqsubseteq C\}}{\mathcal{O}[D/C]} \quad \text{provided } D \notin \mathit{sig}(C)$$

Definer purification:

$$\frac{\mathcal{O}}{\mathcal{O}[D/\top]} \quad \text{provided } D \text{ occurs only positively in } \mathcal{O}$$

Cyclic definer elimination:

$$\frac{\mathcal{O} \cup \{D \sqsubseteq C[D]\}}{\mathcal{O}[D/\nu X.C[X]]} \quad \text{provided } D \in \mathit{sig}(C[D])$$

Figure 4.2: Rewrite rules to eliminate definers from normal clause sets.

Proof. It is easy to see that the declausification rule is equivalence preserving. Note that for normal $\mathcal{ALC}$ clauses, the declausification rule only produces concept inclusions in which no definer occurs negatively on the right-hand side.

For the remaining rules, we have to show that entailments that are not using definer symbols are preserved. In second-order logic, we can eliminate any predicate D from a formula F by replacing it with a quantified variable: $F' = \exists X.F[D/X]$. $F' \models G$ iff $F \models G$ and D does not occur in G (see also Section 2.1). If a second-order formula is of a particular form, existentially quantified predicate variables can be eliminated using Ackermann's Lemma or generalised Ackermann's Lemma (see Section 2.2). The last three rules in Figure 4.2 treat definers as existentially quantified predicate variables, and make use of these lemmata to eliminate them.

For the non-cyclic definer elimination rule and the purification rule, we recall the relevant equivalence of Ackermann's Lemma.

$$\exists X.\forall \vec{x}\big(\neg X(\vec{x}) \vee F(\vec{x}, \vec{y})\big) \wedge G_p[X] \equiv G_p[X/F(\vec{x}, \vec{y})],$$

where X only occurs positively in G_p. If we consider first-order representations of the non-cyclic definer elimination rule, we observe that D only occurs positively in $\mathcal{O}$, and

that $D \sqsubseteq C$ can be expressed in a form similar to the disjunction on the left in the equation. It is then easy to see that the non-cyclic definer elimination rule corresponds to an instance of the equivalence expressed by Ackermann's Lemma.

For the definer purification rule, we adapt the above equivalence by replacing $F(\vec{x}, \vec{y})$ with **true**.

$$\exists X.\forall \vec{x}\big(\neg X(\vec{x}) \vee \mathbf{true}\big) \wedge G_p[X] \equiv G_p[X/\mathbf{true}],$$

The disjunction on the left is tautological. Therefore, Ackermann's Lemma justifies replacing existentially quantified predicate variables by **true**, if they occur only positively. This is what the definer purification rule does.

For the cyclic definer elimination rule, we recall the relevant equivalence of generalised Ackermann's Lemma:

$$\exists X.\forall \vec{x}\big(\neg X(\vec{x}) \vee F_p[X]\big) \wedge G_p[X] \equiv G_p[X/\nu X(\vec{x}).F_p[X])],$$

where X only occurs positively in F_p and G_p. We have the same correspondence between the cyclic definer elimination rule and generalised Ackermann's Lemma as we had between the non-cyclic definer elimination rule and Ackermann's Lemma, and therefore the rule preserves all entailments that do not use the concept symbol D.

We obtain that the result of applying the rules from Figure 4.2 on a set $\mathcal{N}_n$ of normal clauses is equivalent to the second-order formula $\exists \vec{D}.\mathcal{N}_n$, where D ranges over all definers occurring in $\mathcal{N}_n$. Therefore, for every axiom α with $sig(\alpha) \cap N_d = \emptyset$, $\mathcal{O} \models \alpha$ iff $\mathcal{N}_n \models \alpha$. □

4.1.3 Approximating Clause Sets into $\mathcal{ALC}$

Depending on the application, $\mathcal{ALC}\nu$ is not always a suitable representation. State-of-the-art reasoners, as well as the OWL standard, do not support description logics with fixpoint expressions. Moreover, the semantics of fixpoint expressions is not intuitive for all types of end-users of ontologies.

We can obtain a representation without fixpoints by omitting the cyclic definer elimination rule in Figure 4.2 when eliminating definers. The remaining definer symbols then serve as *auxiliary concepts*, which help preserving entailments of cyclic relations that could otherwise not be directly represented in $\mathcal{ALC}$.

The original definition of uniform interpolants requires them to be completely in the desired signature, but $\mathcal{ALC}$ is not expressive enough to represent every uniform interpolant finitely. For this reason, we introduce a generalised definition that allows for finite representations of uniform interpolants in $\mathcal{ALC}$ using auxiliary concepts.

Definition 4.1.6. Let $\mathcal{O}$ be an ontology and $\mathcal{S}$ any signature. $\mathcal{O}$ is an $\mathcal{S}$ *ontology modulo direct cycles* if for every concept symbol $A \notin \mathcal{S}$ there is exactly one axiom of the form $A \sqsubseteq C[A]$, where $C[A]$ contains A, but only positively and nested under role restrictions, and A occurs only positively in the rest of the ontology.

$\mathcal{O}'$ is an $\mathcal{L}$ *uniform interpolant modulo direct cycles* of $\mathcal{O}$ for $\mathcal{S}$ if it is an $\mathcal{S}$ ontology modulo direct cycles and if for any $\mathcal{L}$ axiom α with $sig(\alpha) \subseteq \mathcal{S}$, $\mathcal{O}' \models \alpha$ iff $\mathcal{O} \models \alpha$.

If uniform interpolation is used for computing the logical difference between two ontologies (see Section 1.1), the uniform interpolant has to be both completely in the desired signature and processable by a reasoner. For applications like this, the only option may be to approximate the uniform interpolant. A definer based representation of the uniform interpolant facilitates the computation of these approximations.

Given an $\mathcal{L}$ uniform interpolant modulo direct cycles $\mathcal{O}$ for a signature $\mathcal{S}$, where N'_d is the set of cyclic concept symbols in $\mathcal{S} \setminus sig(\mathcal{O})$, and some positive integer value n representing the approximation depth, we can compute an approximation of the $\mathcal{ALC}$ uniform interpolant using the following algorithm:

1. Set $\mathcal{O}_0 = \mathcal{O} \setminus \{D \sqsubseteq C \mid D \in N'_d\}$.
2. For each i in $1, \ldots, n$:
   (a) Set $\mathcal{O}_i := \mathcal{O}_{i-1}$.
   (b) For each $D \sqsubseteq C \in \mathcal{O}$ with $D \in N'_d$:
      – Set $\mathcal{O}_i := \mathcal{O}_i[D/C]$.
3. For each $D \sqsubseteq C \in \mathcal{O}$ with $D \notin \mathcal{S}$:
   – Set $\mathcal{O}_n^{\mathcal{S}} := \mathcal{O}_n[D/\top]$.

Observe that by definition of $\mathcal{L}$ ontologies modulo direct cycles, every concept D with $D \notin \mathcal{S}$ has exactly one axiom of the form $D \sqsubseteq C$. In Step 1 of the algorithm, we

remove all axioms of this form from the ontology. We then stepwise approximate the uniform interpolant by doing n replacements for each concept D with $D \notin \mathcal{S}$. These replacements are done by expanding the definitions of D, that is, the unique concept inclusion axiom $D \sqsubseteq C$ we removed in the first step. This step is similar to what is done by the non-cyclic definer elimination rule in Figure 4.2, only that D occurs positively in C. Therefore, the result of this transformation still contains the definer D. Observe that D is only allowed to occur positively in the remaining ontology. Therefore, this replacement always ensures $\mathcal{O}_i \models \alpha$ if $\mathcal{O}_{i-1} \models \alpha$ for all $i > 0$ and α with $sig(\alpha) \in \mathcal{S}$. In other words, each incrementation step preserves more entailments of the uniform interpolant. In the last step, we eliminate all symbols $D \notin \mathcal{S}$ by replacing them by $\top$. Since D occurs only positively in the ontology, this result is a lower approximation of the uniform interpolant, and $\mathcal{O}^{\mathcal{S}} \models \mathcal{O}_n$, where $\mathcal{O}^{\mathcal{S}}$ is the uniform interpolant and $\mathcal{O}_n$ the approximation.

4.2 The Refutation Calculus

Our method for computing uniform interpolants is based on a new sound and refutationally complete reasoning method for $\mathcal{ALC}$ ontologies. This method is based on a calculus that infers entailments using a set of rules. In this section, we present this calculus and prove that it is terminating, sound and refutationally complete. These properties come in handy in the next section, where the calculus is extended to a method for computing uniform interpolants of $\mathcal{ALC}\nu$ ontologies. Soundness and refutational completeness of the reasoning method are used later to prove that the uniform interpolation method works correctly.

The calculus introduces new symbols dynamically, which is why we have to adapt the classical notion of soundness. Given a calculus Calc and a set of clauses $\mathcal{N}$, we refer to the saturation of $\mathcal{N}$ using Calc as $\mathsf{Calc}(\mathcal{N})$. According to the traditional notion of soundness, a calculus working on sets of clauses is sound if it only infers clauses that are entailed by the initial clause set. In our context, it is sufficient to require that inferences do not lead to new entailments modulo definer symbols.

Definition 4.2.1. Let Calc be a calculus. Calc is *sound* if for any set of clauses $\mathcal{N}$ and any axiom α with $sig(\alpha) \cap N_d = \emptyset$, $\mathsf{Calc}(\mathcal{N}) \models \alpha$ only if $\mathcal{N} \models \alpha$. Calc is *refutationally*

Resolution

$$\frac{C_1 \sqcup A \qquad C_2 \sqcup \neg A}{C_1 \sqcup C_2}$$

$\forall\exists$-Role Propagation

$$\frac{C_1 \sqcup \forall r.D_1 \qquad C_2 \sqcup \exists r.D_2}{C_1 \sqcup C_2 \sqcup \exists r.D_{12}}$$

where D_{12} is a possibly new definer representing $D_1 \sqcap D_2$.

$\exists$-Elimination

$$\frac{C_1 \sqcup \exists r.D \qquad \neg D}{C_1}$$

Figure 4.3: Rules of calculus $Res_{\mathcal{ALC}}$.

complete if whenever $\bot \notin \mathsf{Calc}(\mathcal{N})$, $\mathcal{N}$ is satisfiable. Calc is *terminating* if $\mathsf{Calc}(\mathcal{N})$ is a finite set for any finite set of clauses $\mathcal{N}$.

4.2.1 The Rules of the Calculus

$Res_{\mathcal{ALC}}$ is the refutation calculus that underlies the uniform interpolation method presented in this chapter. It is applied to $\mathcal{ALC}\nu$ ontologies in normal form, and uses the rules that are shown in Figure 4.3.

All three rules, and most of the rules presented in the following chapters, are based on the valid concept inclusion $(C_1 \sqcup C_2) \sqcap (C_3 \sqcup C_4) \sqsubseteq C_1 \sqcup C_3 \sqcup (C_2 \sqcap C_4)$ (see Figure 4.4 for a visual illustration of this). Due to this inclusion, any rule is sound if its premises are of the form $C_1 \sqcup L_1$ and $C_2 \sqcup L_2$, and its conclusion is entailed by $C_1 \sqcup C_2 \sqcup (L_1 \sqcap L_2)$. For example, the *resolution rule* is sound because $(A \sqcap \neg A) \sqsubseteq \bot$ is a valid concept inclusion, and so is $C_1 \sqcup C_2 \sqcup (A \sqcap \neg A) \sqsubseteq C_1 \sqcup C_2$.

The same idea is followed by the $\forall\exists$*-role propagation rule*. For role restrictions, we have the valid concept inclusion $(\forall r.D_1 \sqcap \exists r.D_2) \sqsubseteq \exists r.(D_1 \sqcap D_2)$. $\exists r.(D_1 \sqcap D_2)$ is not an $\mathcal{ALC}$ literal, since it contains a conjunction nested under a role restriction. For this reason, the calculus introduces definer symbols dynamically.

A definer symbol D_{12} representing $D_1 \sqcap D_2$ is introduced by adding the two clauses $\neg D_{12} \sqcup D_1$ and $\neg D_{12} \sqcup D_2$ to the current clause set. These two clauses are

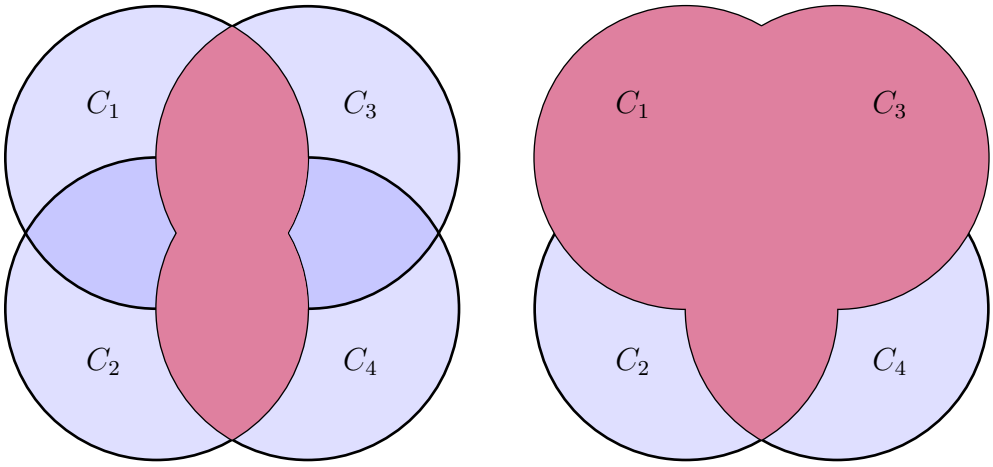

Figure 4.4: Venn diagram illustrating $\models (C_1 \sqcup C_2) \sqcap (C_3 \sqcup C_4) \sqsubseteq C_1 \sqcup C_3 \sqcup (C_2 \sqcap C_4)$.

equivalent to the TBox axiom $D_{12} \sqsubseteq D_1 \sqcap D_2$. In order to obtain a terminating procedure, we keep track of introduced definers, and reuse previously introduced definers whenever possible. This is described in more detail in Section 4.2.2. In the remainder of the thesis we keep the convention that for two definers D_a and D_b, we denote the definer representing $D_a \sqcap D_b$ by D_{ab}, using the index to indicate which conjunction it represents.

The unary clause $\neg D$ expresses that the definer D is unsatisfiable. If D is unsatisfiable, concepts of the form $\exists r.D$ are unsatisfiable as well. No domain element can have an r-successor satisfying D, if D itself is unsatisfiable. Therefore, whenever we have a unary clause of the form $\neg D$, we can use the $\exists$-*elimination rule* to eliminate literals of the form $\exists r.D$ from a clause.

4.2.2 Introducing Definers

In order to obtain a terminating calculus, that is, in order to ensure that only a finite number of clauses can be derived from any finite set of clauses, we introduce definer symbols only if necessary. This is done using an updated mapping $conj : N_d \mapsto 2^{N_d^*}$ that maps definers to sets of *base definers*. The set N_d^* of base definers denotes the definer symbols that are present in the input clause set. These are the definer symbols that have been introduced by the normal form transformation. Intuitively, *conj* maps

each definer to the set of base definers it represents. In other words, if $conj(D) = \{D_1, \ldots, D_n\}$, then $Res_{\mathcal{ALC}}(\mathcal{N}) \models D \sqsubseteq D_1 \sqcap \ldots \sqcap D_n$.

Initially, *conj* maps every base definer $D_b \in N_d^*$ to the singleton set $\{D_b\}$ consisting just of the base definer itself. If a definer D_{12} representing the conjunction $D_1 \sqcap D_2$ is part of the conclusion of a rule application, we check whether there is a definer D'_{12} in our clause set such that $conj(D'_{12}) = conj(D_1) \cup conj(D_2)$. If yes, we reuse D'_{12} to represent $D_1 \sqcap D_2$. Additionally, we add the two clauses $\neg D'_{12} \sqcup D_1$ and $\neg D'_{12} \sqcup D_2$ to the current clause set, if they are not already present.

If there is no definer D'_{12} with $conj(D'_{12}) = conj(D_1) \cup conj(D_2)$ in the current clause set, we introduce D_{12} as new definer and add the two clauses $\neg D_{12} \sqcup D_1$ and $\neg D_{12} \sqcup D_2$ to the current clause set. Observe that this way, *conj* is always ensured to be an injective function over the introduced definers.

Lemma 4.2.2. *The $\forall\exists$-role propagation rule using the described technique for reusing definer symbols is sound.*

Proof. The $\forall\exists$-role propagation rule inferes from two clauses $C_1 = C'_1 \sqcup \forall r.D_1$ and $C_2 = C'_2 \sqcup \exists r.D_2 \in \mathcal{N}^*$ the conclusion $C = C'_1 \sqcup C'_2 \sqcup \exists r.D_{12}$. Suppose $conj(D_1) = \{D_{11}, \ldots, D_{1n}\}$ and $conj(D_2) = \{D_{21}, \ldots, D_{2m}\}$. Regardless of whether D_{12} has been reused or is freshly introduced, $conj(D_{12}) = conj(D_1) \cup conj(D_2)$. By Theorem 4.1.5, eliminating definers as discussed in Section 4.1.2 preserves all entailments α with $sig(\alpha) \cap N_d = \emptyset$. Eliminating all introduced definers in the two premises results in $C'_1 \sqcup \forall r.(D_{11} \sqcap \ldots \sqcap D_{1n})$ and $C'_2 \sqcup \exists r.(D_{21} \sqcap \ldots \sqcap D_{2m})$, and eliminating all introduced definers in the conclusion C results in $C' = C'_1 \sqcup C'_2 \sqcup \exists r.(D_{11} \sqcap \ldots \sqcap D_{1n} \sqcap D_{21} \sqcap \ldots \sqcap D_{2m})$. This conclusion is a logical consequence of the transformed premises. □

Lemma 4.2.3. *The calculus $Res_{\mathcal{ALC}}$ is sound.*

Proof. We established soundness of the role propagation rule using our technique for reusing definer symbols in Lemma 4.2.2. The soundness of the remaining rules has been discussed in Section 4.2.1. □

Lemma 4.2.4. *$Res_{\mathcal{ALC}}$ is terminating, and the size of $Res_{\mathcal{ALC}}(\mathcal{N})$ is bounded by $O(2^{2^n} \times 2^n)$, where $n = size(\mathcal{N})$.*

Proof. Let $\mathcal{N}$ be any set of clauses and $n = size(\mathcal{N})$. Denote by N_c^*, N_r^* and N_d^* respectively the concept, role and definer symbols occurring in $\mathcal{N}$. Note that $|N_c^*|$, $|N_r^*|$ and $|N_d^*|$ are linear in n. Every definer in $Res_{\mathcal{ALC}}(\mathcal{N})$ occurs in the domain of *conj*. Since *conj* is injective, and the codomain of *conj* is $2^{N_d^*}$, $Res_{\mathcal{ALC}}(\mathcal{N})$ contains at most $n_d = 2^{|N_d^*|}$ many definer symbols.

Every concept symbol can occur either negatively or positively in a literal, and every definer symbol can occur in a literal of the forms D, $\neg D$, $\exists r.D$ and $\forall r.D$. We obtain the following upper bound for the number of distinct literals in $Res_{\mathcal{ALC}}(\mathcal{N})$:

$$2 \times N_c^* + 2 \times n_d + 2 \times N_r^* \times n_d \tag{4.12}$$

$$= 2 \times N_c^* + 2 \times 2^{|N_d^*|} + 2 \times N_r^* \times 2^{|N_d^*|} \tag{4.13}$$

$$= O(2^n) \tag{4.14}$$

Since clauses are represented as sets, we can represent maximally $O(2^{2^n})$ different clauses using these literals, which means $Res_{\mathcal{ALC}}(\mathcal{N})$ contains at most $O(2^{2^n})$ clauses that contain at most $O(2^n)$ literals each. Therefore, the size of $Res_{\mathcal{ALC}}(\mathcal{N})$ is bounded by $O(2^{2^n} \times 2^n)$. □

Because of how definers are introduced, we have the following lemma regarding positive definer literals that will become useful later on.

Lemma 4.2.5. *Let $\mathcal{N} = Cl(\mathcal{O})$ be the normal form representation of the $\mathcal{ALC}$ ontology $\mathcal{O}$, and let $C \in Res_{\mathcal{ALC}}(\mathcal{N})$ be a clause that contains a positive definer literal D. Then, C is of the form $\neg D' \sqcup D$, $D' \in N_d$.*

Proof. No clause in $\mathcal{N}$ contains any positive definer literals. Therefore, every clause in $\mathcal{N}$ that contains a negative definer literal is of the form $\neg D \sqcup C$, where C does not contain any positive definer literals. New definers are introduced together with clauses of the form $\neg D_1 \sqcup D_2$, $D_1, D_2 \in N_d$. We show that, if we have a set of clauses such that every clause is of the forms $\neg D \sqcup C$, C and $\neg D_1 \sqcup D_2$, where C does not contain any positive definer literals, it is impossible to derive a clause which is not of the same form.

1. Assume both premises of a rule are of the form $\neg D \sqcup C$ or C, where C does not contain any positive definer literal. Since the premises do not contain positive definer literals, the conclusion also does not contain a positive definer literal.

2. Assume one premise of a rule is of the form $\neg D_1 \sqcup D_2$. The only rule that applies to this clause is the resolution rule, which means the other premise must either be of the form $\neg D_2 \sqcup C$, $\neg D_2 \sqcup D_3$ or $\neg D_3 \sqcup D_1$, where C does not contain any positive definer literals.

In both cases, the resolvent either does not contain a positive definer literal, or it is of the form $\neg D_1' \sqcup D_2'$. Therefore, no sequence of inferences can infer a clause that contains a positive definer literal and is of a different form. We obtain that every clause in $Res_{\mathcal{ALC}}(\mathcal{N})$ either does not contain any positive definer literal, or it is of the form $\neg D_1 \sqcup D_2$. □

Example 4.2.6. Let $\mathcal{O}_2$ be the following ontology:

$$\top \sqsubseteq \exists r.A \qquad A \sqsubseteq \forall r.\neg A$$

We want to decide whether $\mathcal{O}_2$ is satisfiable. First, we compute the normal form representation $\mathcal{N}_2 = Cl(\mathcal{O}_2)$ of the ontology.

1. $\exists r.D_1$
2. $\neg D_1 \sqcup A$
3. $\neg A \sqcup \forall r.D_2$
4. $\neg D_2 \sqcup \neg A$

Initially, the mapping *conj* maps each introduced definer to a singleton set containing itself.

$$conj(D_1) = \{D_1\}$$
$$conj(D_2) = \{D_2\}$$

We can apply $\forall\exists$-role propagation on Clause 1 and Clause 3. The conclusion of this rule requires a definer D_{12} representing $D_1 \sqcap D_2$. For this definer D_{12}, we should have $conj(D_{12}) = conj(D_1) \cup conj(D_2) = \{D_1, D_2\}$. Such a definer does not exist, so we introduce it.

5. $\neg A \sqcup \exists r.D_{12}$ *(Role propagation 1, 3)*
6. $\neg D_{12} \sqcup D_1$ $(D_{12} \sqsubseteq D_1)$
7. $\neg D_{12} \sqcup D_2$ $(D_{12} \sqsubseteq D_2)$

We set $conj(D_{12}) = \{D_1, D_2\}$. Another application of the role propagation rule on Clauses 1 and 3 would reuse the definer D_{12}, and therefore no different clause would be inferred. We continue by applying resolution on the positive definer literals.

8. $\neg D_{12} \sqcup A$ *(Resolution 2, 6)*
9. $\neg D_{12} \sqcup \neg A$ *(Resolution 4, 5)*
10. $\neg D_{12}$ *(Resolution 8, 9)*

The $\exists$-elimination rule can now be applied twice, resulting in a derivation of the empty clause.

11. $\neg A$ *($\exists$-Elimination 5, 10)*
12. $\neg D_1$ *(Resolution 2, 12)*
13. $\bot$ *($\exists$-Elimination 1, 12)*

We have inferred that $\mathcal{O}_2$ is unsatisfiable.

4.2.3 Refutational Completeness

In the last section, we proved soundness and termination of $Res_{\mathcal{ALC}}$. In order to prove refutational completeness, we have to show that whenever $Res_{\mathcal{ALC}}(\mathcal{N})$ does not contain the empty clause, $\mathcal{N}$ is satisfiable. We prove this by showing how, given $\bot \notin Res_{\mathcal{ALC}}(\mathcal{N})$, a model for $\mathcal{N}$ can be constructed based on the clauses in $Res_{\mathcal{ALC}}(\mathcal{N})$. The proof is influenced by the model construction used in Bachmair and Ganzinger (2001) to prove refutational completeness of propositional resolution.

Let $\mathcal{N}$ be any set of $\mathcal{ALC}$ clauses, $\mathcal{N}^* = Res_{\mathcal{ALC}}(\mathcal{N})$ its saturation, and $\bot \notin \mathcal{N}^*$.

We first define an ordering on the definer symbols in $\mathcal{N}^*$. Let $\prec_d$ be an ordering such that $D_1 \prec_d D_2$ if $\neg D_1 \sqcup D_2 \in \mathcal{N}^*$. Intuitively, this ordering represents the subsumption hierarchy between introduced definers, in the sense that $\mathcal{N}^* \models D_1 \sqsubseteq D_2$ implies $D_1 \prec_d D_2$. Note that a definer D_{12} representing $D_1 \sqcap D_2$ is introduced by adding the two clauses $\neg D_{12} \sqcup D_1$ and $\neg D_{12} \sqcup D_2$. The ordering therefore ensures that for every introduced definer D_{12} that represents $D_1 \sqcap D_2$, $D_{12} \prec_d D_1$ and $D_{12} \prec_d D_2$. There is always an ordering $\prec_d$ that satisfies these properties, as the following lemma shows.

Lemma 4.2.7. *Let $\prec_d$ be a relation between definers such that $\neg D_1 \sqcup D_2 \in \mathcal{N}^*$ iff $D_1 \prec_d D_2$. Then, $\prec_d$ is transitive and irreflexive.*

Proof. Transitivity follows from the fact that $\mathcal{N}^*$ is closed under resolution: $\neg D_1 \sqcup D_2$, $\neg D_2 \sqcup D_3 \in \mathcal{N}^*$ implies $\neg D_1 \sqcup D_3 \in \mathcal{N}^*$, due to resolution on D_2. Irreflexivity follows from how definers are introduced. The result of the normal form transformation does not contain any clauses of the form $\neg D_1 \sqcup D_2$. Clauses of this form are only introduced if the definer D_1 is freshly introduced and D_2 is already present. Hence, $\prec_d$ cannot have cycles, and is therefore irreflexive. □

Let $\prec_s$ be any total ordering on the symbols $(N_c \cup N_r) \setminus N_d$. Based on $\prec_d$ and $\prec_s$, we define a total ordering $\prec_l$ on literals that satisfies the following constraints.

1. $A \prec_l \neg A$ for all $A \in N_c$.
2. $\exists r.D_1 \prec_l \forall r.D_2$ for all $r \in N_r$ and $D_1, D_2 \in N_d$.
3. $\neg D \prec_l A$ and $\neg D \prec_l \exists r.D'$, for all $D, D' \in N_d$, $A \in N_c \setminus N_d$ and $r \in N_r$.
4. If $D_1 \prec_d D_2$, then $\neg D_1 \prec_l D_2$ and $\exists r.D_1 \prec_l \exists r.D_2$.
5. If $A_1 \prec_s A_2$, then $\neg A_1 \prec_l A_2$.
6. If $A \prec_s r$, then $\neg A \prec_l \exists r.D$ for all $D \in N_d$.
7. If $r \prec_s A$, then $\forall r.D \prec_l A$ for all $D \in N_d$.
8. If $r_1 \prec_s r_2$, then $\forall r_1.D_1 \prec_l \exists r_2.D_2$ for all $D_1, D_2 \in N_d$.

That an ordering with these constraints always exists, is shown below. We first give an intuitive idea of what the constraints express.

Constraints 1 and 2 determine the order between literals if the concept or role symbol in the literal is the same. In the ordering, negative literals are larger than positive literals, and universal role restrictions are larger than existential role restrictions. Constraint 3 states that definer literals are smaller than other literals. Between literals of the same type (definer literals, concept literals, role restrictions), the ordering is determined by $\prec_d$ and $\prec_s$, as stated in the remaining constraints. Constraints 6, 7 and 8 are not strictly needed for the completeness proof. However, the fact that the

ordering is determined by a total ordering $\prec_s$ on symbols in $(N_c \cup N_r) \setminus N_d$ is used in Section 4.4 for showing the correctness of the uniform interpolation method.

How the ordering works is illustrated in the following running example.

Example 4.2.8. Let $\mathcal{O}_3$ be the following ontology.

$$\top \sqsubseteq A \sqcup B$$

$$B \sqsubseteq \exists r.C$$

$$C \sqsubseteq \forall r.A$$

The saturation $\mathcal{N}_3^* = Res_{\mathcal{ALC}}(\mathcal{N}_3)$, where $\mathcal{N}_3 = Cl(\mathcal{O}_3)$, can be seen in Figure 4.5. We follow the steps to create an ordering $\prec_l$ for the literals in $\mathcal{N}_3^*$.

The ordering $\prec_d$ on definer symbols has to obey $D_{12} \prec_d D_1$ and $D_{12} \prec_d D_2$ due to Clause 8 and 9. This can be extended to the following total ordering:

$$D_{12} \prec_d D_1 \prec_d D_2$$

We fix the following ordering $\prec_s$ on the symbols in $(N_c \cup N_r) \setminus N_d$.

$$A \prec_s B \prec_s C \prec_s r$$

Based on $\prec_s$ and $\prec_d$, we set the following ordering $\prec_l$ on literals:

$$\begin{aligned} D_{12} \prec_l \neg D_{12} \prec_l D_1 \prec_l \neg D_1 \prec_l D_2 \prec_l \neg D_2 \\ \prec_l A \prec_l \neg A \prec_l B \prec_l \neg B \prec_l C \prec_l \neg C \\ \prec_l \exists r.D_{12} \prec_l \exists r.D_1 \prec_l \exists r.D_2 \\ \prec_l \forall r.D_{12} \prec_l \forall r.D_1 \prec_l \forall r.D_2 \end{aligned}$$

One can verify that $\prec_l$ satisfies all required constraints, which in this case are the following:

Constraint 1:	$D_1 \prec_l \neg D_1, D_2 \prec_l \neg D_2, D_{12} \prec_l \neg D_{12}$
	$A \prec_l \neg A, B \prec_l \neg B, C \prec_l \neg C$
Constraint 2:	$\exists r.D_1 \prec_l \forall r.D_1, \forall r.D_2, \forall r.D_{12}$
	$\exists r.D_2 \prec_l \forall r.D_1, \forall r.D_2, \forall r.D_{12}$
	$\exists r.D_{12} \prec \forall r.D_1, \forall r.D_2, \forall r.D_{12}$

1. $A \sqcup B$	$(\mathcal{N} = Cl(\mathcal{O}_3))$
2. $\neg B \sqcup \exists r.D_1$	$(\mathcal{N} = Cl(\mathcal{O}_3))$
3. $\neg D_1 \sqcup C$	$(\mathcal{N} = Cl(\mathcal{O}_3))$
4. $\neg C \sqcup \forall r.D_2$	$(\mathcal{N} = Cl(\mathcal{O}_3))$
5. $\neg D_2 \sqcup A$	$(\mathcal{N} = Cl(\mathcal{O}_3))$
6. $A \sqcup \exists r.D_1$	*(Resolution 1, 2, B)*
7. $\neg B \sqcup \neg C \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 2, 4)*
8. $\neg D_{12} \sqcup D_1$	$(D_{12} \sqsubseteq D_1)$
9. $\neg D_{12} \sqcup D_2$	$(D_{12} \sqsubseteq D_2)$
10. $\neg D_1 \sqcup \forall r.D_2$	*(Resolution 3, 4, C)*
11. $\neg D_1 \sqcup \neg B \sqcup \exists r.D_{12}$	*(Resolution 3, 7, C)*
12. $\neg D_{12} \sqcup C$	*(Resolution 3, 8, D_1)*
13. $A \sqcup \neg C \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 4, 6)*
14. $\neg D_1 \sqcup \neg B \sqcup \neg C \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 4, 11)*
15. $\neg D_{12} \sqcup \forall r.D_2$	*(Resolution 4, 12, C)*
16. $\neg D_{12} \sqcup A$	*(Resolution 5, 9, D_2)*
17. $\neg D_1 \sqcup A \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 6, 10)*
18. $\neg D_{12} \sqcup A \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 6, 15)*
19. $\neg D_{12} \sqcup \neg B \sqcup \exists r.D_{12}$	*(Resolution 7, 12, C)*
20. $\neg D_{12} \sqcup \neg B \sqcup \neg C \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 7, 15)*
21. $\neg D_1 \sqcup A \sqcup \neg C \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 10, 13)*
22. $\neg D_{12} \sqcup \neg D_1 \sqcup A \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 10, 18)*
23. $\neg D_{12} \sqcup \neg D_1 \sqcup \neg B \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 10, 19)*
24. $\neg D_{12} \sqcup \neg D_1 \sqcup \neg B \sqcup \neg C \sqcup \exists r.D_{12}$	*(∀∃-Role Propagation 10, 20)*

Figure 4.5: Saturation $\mathcal{N}_3^*$ of the clausal representation $\mathcal{N}_3$ of the satisfiable example ontology $\mathcal{O}_3$.

Constraint 3:	$\neg D_1 \prec_l A, B, C, \exists r.D_1, \exists r.D_2, \exists r.D_{12}$
	$\neg D_2 \prec_l A, B, C, \exists r.D_1, \exists r.D_2, \exists r.D_{12}$
	$\neg D_{12} \prec_l A, B, C, \exists r.D_1, \exists r.D_2 \exists r.D_{12}$
Constraint 4:	$\neg D_{12} \prec_l D_1, \neg D_1 \prec_l D_2$
	$\exists r.D_{12} \prec_l \exists r.D_1 \prec_l \exists r.D_2$
Constraint 5:	$\neg A \prec_l B, \neg B \prec_l C$
Constraint 6:	$A, B, C \prec_l \exists r.D_1, \exists r.D_2, \exists r.D_{12}$

In order to determine the ordering between two literals, we first compare the concept symbol or role symbol, depending on whether the literal is a role restriction, then the operator (negation, existential role restriction or universal role restriction), and then, if the literal is a role restriction, we compare the definer symbol under the role restriction. The ordering makes furthermore sure that definer literals are always smaller than other literals.

If we represent literals as triples, $\prec_l$ corresponds to a lexicographic ordering. Let $\prec_x$ be a total ordering on symbols in $N_c \cup N_r \cup \{\varepsilon, \neg, \exists, \forall\}$ that extends $\prec_s$ and $\prec_d$, such that $\varepsilon \prec_x \neg$, $\exists \prec_x \forall$ and definer symbols are smaller than other symbols. Let t be a function that translates literals into triples according to the following rules.

$$t(A) = \langle A, \varepsilon, \varepsilon \rangle$$
$$t(\neg A) = \langle A, \neg, \varepsilon \rangle$$
$$t(\exists r.D) = \langle r, \exists, D \rangle$$
$$t(\forall r.D) = \langle r, \forall, D \rangle$$

$\prec_l$ corresponds to the lexicographic ordering on the triple representations of the literals: $L_1 \prec_l L_2$ iff $t(L_1) \prec_x^3 t(L_2)$, where $\prec_x^3$ is the ternary lexicographic extension of $\prec_x$. If we look at the triple representations of the literals in our example, we can easily see that the total ordering we gave corresponds to a lexicographic ordering:

$\langle D_{12}, \varepsilon, \varepsilon \rangle$,	$\langle D_2, \varepsilon, \varepsilon \rangle$,	$\langle B, \varepsilon, \varepsilon \rangle$,	$\langle r, \exists, D_{12} \rangle$,	$\langle r, \forall, D_1 \rangle$,
$\langle D_{12}, \neg, \varepsilon \rangle$,	$\langle D_2, \neg, \varepsilon \rangle$,	$\langle B, \neg, \varepsilon \rangle$,	$\langle r, \exists, D_1 \rangle$,	$\langle r, \forall, D_2 \rangle$.
$\langle D_1, \varepsilon, \varepsilon \rangle$,	$\langle A, \varepsilon, \varepsilon \rangle$,	$\langle C, \varepsilon, \varepsilon \rangle$,	$\langle r, \exists, D_2 \rangle$,	
$\langle D_1, \neg, \varepsilon \rangle$,	$\langle A, \neg, \varepsilon \rangle$,	$\langle C, \neg, \varepsilon \rangle$,	$\langle r, \forall, D_{12} \rangle$,	

Lemma 4.2.9. *Given the orderings $\prec_d$ and $\prec_s$, there is always an ordering $\prec_l$ on literals that satisfies Constraints 1–8.*

Proof. Let $\prec_l$ be an ordering such that $L_1 \prec_l L_2$ iff $t(L_1) \prec_x^3 t(L_2)$. We show that $\prec_l$ satisfies all constraints. Constraint 1 is satisfied since $\varepsilon \prec_x \neg$. Constraint 2 is satisfied because $\exists \prec_x \forall$. Constraints 3 and 4 are satisfied because $\prec_x$ makes definer symbols smaller than other symbols. Constraints 5–8 correspond to the cases where the triple representations start with a different symbol. □

A literal L is *maximal* in a clause C if for all literals $L' \in C$ with $L \neq L'$, $L' \prec_l L$.

$\prec_l$ is extended to an ordering $\prec_c$ between clauses using the multiset extension of $\prec_l$, that is, $C_1 \prec_c C_2$ if there is a literal $L_2 \in (C_2 \setminus C_1)$ such that $L_1 \prec_l L_2$ for every literal $L_1 \in (C_1 \setminus C_2)$. This ordering allows us to enumerate the clauses in $\mathcal{N}^*$. In the following, we denote by C_i the ith clause in $\mathcal{N}^*$ according to this enumeration, that is, $C_i \prec_c C_k$ implies $i < k$.

Example 4.2.10. Figure 4.6 lists the clauses of $\mathcal{N}_3^*$ ordered with respect to $\prec_c$, where the maximal literal in each clause is in bold face. We inverted the order in which the literals are displayed, starting from the largest instead of the smallest, to visualise more clearly how the ordering is implemented.

The model construction now proceeds as follows. We build a candidate model for $\mathcal{N}^*$ based on a number of *model fragments*, where each definer symbol D is represented by a model fragment I^D. For each model fragment, the final candidate model will have a corresponding domain element. For a definer D, the model fragment I^D contains the concept symbols and existential restrictions a domain element has to satisfy if it satisfies the definer D.

Definition 4.2.11. A *model fragment* is a set of positive literals of the forms A and $\exists r.D$, where $A \in N_c$, $r \in N_r$ and $D \in N_d$. Given a model fragment I and a literal L, I *satisfies* L, in symbols $I \models L$, if one of the following conditions hold:

1. $L = A$, $A \in N_c$, and $L \in I$.
2. $L = \neg A$, $A \in N_c$, and $L \notin I$.
3. $L = \exists r.D$, $r \in N_r$, $D \in N_d$, and $\exists r.D' \in I$ where either $D' = D$ or $\neg D' \sqcup D \in \mathcal{N}^*$.

$$
\begin{aligned}
C_1 &= \mathbf{D_1} \sqcup \neg D_{12} && \textit{(Clause 8)}\\
C_2 &= \mathbf{D_2} \sqcup \neg D_{12} && \textit{(Clause 9)}\\
C_3 &= \mathbf{A} \sqcup \neg D_{12} && \textit{(Clause 16)}\\
C_4 &= \mathbf{A} \sqcup \neg D_2 && \textit{(Clause 5)}\\
C_5 &= \mathbf{B} \sqcup A && \textit{(Clause 1)}\\
C_6 &= \mathbf{C} \sqcup \neg D_{12} && \textit{(Clause 12)}\\
C_7 &= \mathbf{C} \sqcup \neg D_1 && \textit{(Clause 3)}\\
C_8 &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup A \sqcup \neg D_{12} && \textit{(Clause 18)}\\
C_9 &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup A \sqcup \neg D_1 && \textit{(Clause 17)}\\
C_{10} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup A \sqcup \neg D_1 \sqcup \neg D_{12} && \textit{(Clause 22)}\\
C_{11} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup \neg B \sqcup \neg D_{12} && \textit{(Clause 19)}\\
C_{12} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup \neg B \sqcup \neg D_1 && \textit{(Clause 11)}\\
C_{13} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup \neg B \sqcup \neg D_1 \sqcup \neg D_{12} && \textit{(Clause 23)}\\
C_{14} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup \neg C \sqcup A && \textit{(Clause 13)}\\
C_{15} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup \neg C \sqcup A \sqcup \neg D_1 && \textit{(Clause 21)}\\
C_{16} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup \neg C \sqcup \neg B && \textit{(Clause 7)}\\
C_{17} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup \neg C \sqcup \neg B \sqcup \neg D_{12} && \textit{(Clause 20)}\\
C_{18} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup \neg C \sqcup \neg B \sqcup \neg D_1 && \textit{(Clause 14)}\\
C_{19} &= \exists \mathbf{r}.\mathbf{D_{12}} \sqcup \neg C \sqcup \neg B \sqcup \neg D_1 \sqcup \neg D_{12} && \textit{(Clause 24)}\\
C_{20} &= \exists \mathbf{r}.\mathbf{D_1} \sqcup A && \textit{(Clause 6)}\\
C_{21} &= \exists \mathbf{r}.\mathbf{D_1} \sqcup \neg B && \textit{(Clause 2)}\\
C_{22} &= \forall \mathbf{r}.\mathbf{D_2} \sqcup \neg D_{12} && \textit{(Clause 15)}\\
C_{23} &= \forall \mathbf{r}.\mathbf{D_2} \sqcup \neg D_1 && \textit{(Clause 10)}\\
C_{24} &= \forall \mathbf{r}.\mathbf{D_2} \sqcup \neg C && \textit{(Clause 4)}
\end{aligned}
$$

Figure 4.6: The enumerated set $\mathcal{N}_3^*$, ordered by $\prec_c$.

4. $L = \forall r.D$, $r \in N_r, D \in N_d$, and there is no $\exists r.D' \in I$ such that $D' \neq D$ and $\neg D' \sqcup D \notin \mathcal{N}^*$.

Given a model fragment I and a clause C, we say I *satisfies* C, in symbols $I \models C$, if there is a literal $L \in C$ such that $I \models L$.

Based on the enumerated clauses, we build a model fragment I^D for each definer D in $\mathcal{N}^*$, plus one model fragment I^ϵ. I^ϵ represents the root of the model, and is represented by a domain element that does not satisfy any definer symbol. It is needed in case $\mathcal{N}^*$ only has models in which no definer is satisfied. For definers $D \in N_d$ with $\neg D \in \mathcal{N}^*$, we set $I^D = \emptyset$. For the remaining definers and ϵ, the model fragment I^D is defined incrementally as follows.

1. $I^D_0 = \emptyset$ if $D = \epsilon$, and $I^D_0 = \{D\}$ if $D \in N_d$.

2. If $I^D_{i-1} \models C_i$, then $I^D_i = I^D_{i-1}$, otherwise:

   (a) If the maximal literal in C_i is of the form A, then $I^D_i = I^D_{i-1} \cup \{A\}$

   (b) If the maximal literal in C_i is of the form $\exists r.D'$, then $I^D_i = I^D_{i-1} \cup \{\exists r.D'\}$

3. $I^D = I^D_n$, where $n = |\mathcal{N}^*|$.

If $\neg D \in \mathcal{N}^*$, then D is unsatisfiable, and we cannot build a model fragment for it. This is why in this case, we leave I^D empty . Otherwise, the model fragment I^D is built in n iterations starting from I^D_0. I^D is supposed to be a model fragment for D, and therefore we initialise it with D if D is a definer, or with the empty set if $D = \epsilon$. In Step 2 we then check for each clause C_i, following the ordering $\prec_c$ represented in the enumeration of clauses, whether it is already satisfied by the current model fragment I^D_{i-1}. If this is not the case, and the maximal literal in C_i is of the form A or $\exists r.D'$, we ensure $I^D_i \models C_i$ by adding this maximal literal to I^D_i.

Example 4.2.12. We initialise four model fragments, the empty I^ϵ_0 and one model fragment for each definer D_1, D_2 and D_{12}. The model construction processes the clauses in the order as they are shown in Figure 4.6, and adds the maximal literal to the model fragment whenever a clause is not satisfied by the current model fragment.

As a result, we obtain the following four model fragments:

$$I^{\epsilon} = \{B, \exists r.D_1\}$$
$$I^{D_1} = \{D_1, B, C, \exists r.D_{12}\}$$
$$I^{D_2} = \{D_2, A\}$$
$$I^{D_{12}} = \{D_{12}, D_1, D_2, A, C\}$$

The first model fragment I^{ϵ} is initialised with the empty set: $I^{\epsilon}_0 = \emptyset$. The smallest clause that is not satisfied by I^{ϵ}_0 is C_5, in which the maximal literal is B. Therefore, we have $I^{\epsilon}_5 = \{B\}$. Most of the following clauses have at least one negative literal that is different from $\neg B$, and are therefore satisfied by I^{ϵ}_5. The next smallest clause that is not satisfied by I^{ϵ}_5 is C_{20}. The maximal literal in C_{20} is $\exists r.D_1$, and therefore $I^{\epsilon}_{20} = \{B, \exists r.D_1\}$. I^{ϵ}_{20} satisfies all larger clauses, and therefore $I^{\epsilon} = I^{\epsilon}_{20}$. The other model fragments all start with a different initial literal set, depending on the definer they represent, and have therefore varying content. Note for I^{D_1}, that $\{\exists r.D_{12}\} \models \exists r.D_1$ in our context, since $\neg D_{12} \sqcup D_1 \in \mathcal{N}^*$. Therefore, $\exists r.D_1$ is not added to this model fragment.

One can verify that every model fragment satisfies its designated definer, as well as all clauses in $\mathcal{N}^*_3$.

Before we show that the construction always results in a model fragment that satisfies all clauses, we prove monotonicity of the model construction, since this property is used in the proofs that follow. Monotonicity has the following meaning in this context: if a clause C_i is satisfied by a model fragment I^D_i, it is also satisfied by all following model fragments I^D_j with $j > i$, and if a clause C_i is not satisfied by I^D_i, it is also not satisfied by all following model fragments I^D_j with $j > i$. Each statement is proved in its own lemma.

Lemma 4.2.13. *If $I^D_i \models C_i$, then $I^D_j \models C_i$ for all $j > i$.*

Proof. Assume $I^D_i \models C_i$ and $j > i$. By definition, C_i must contain at least one literal L with $I^D_i \models L$. We distinguish the different cases for L.

1. Assume L is of the form A or $\exists r.D_1$. In this case, $I^D_i \models C_i$ implies either $L \in I^D_i$ or $L = \exists r.D_1$ and $\exists r.D_2 \in I^D_i$ with $\neg D_2 \sqcup D_1 \in \mathcal{N}^*$. Since the model construction

only adds literals, but does not remove any, $I_i^D \subseteq I_j^D$. Therefore, either $L \in I_j^D$ or $\exists r.D' \in I_j^D$ with $\neg D' \sqcup D \in \mathcal{N}^*$, and hence $I_j^D \models C_i$.

2. Assume L is of the form $\neg A$. We prove $I_j^D \models C_i$ by contradiction and assume $I_j^D \not\models C_i$. $I_i^D \models L$ implies $A \notin I_i^D$ and $I_j^D \not\models L$ implies $A \in I_j^D$. Our model construction only adds maximal literals to the model fragment. Therefore, there must be a clause C_k with $i < k < j$, in which the maximal literal is A. According to our ordering, $A \prec_l \neg A$. But then, since A is maximal in C_k, we have a literal in C_i that is larger than all literals in C_k, namely $\neg A$. This implies $k < i$. We have a contradiction, based on the assumption $I_j^D \not\models C_i$. Therefore, $I_j^D \models C_i$.

3. Assume L is of the form $\forall r.D_1$. Again, we prove $I_j^D \models C_i$ by contradiction and assume $I_j^D \not\models C_i$. If $I_j^D \not\models C_i$, there must be a literal $\exists r.D_2 \in I_j^D$ with $D_2 \neq D_1$ and $\neg D_2 \sqcup D_1 \notin \mathcal{N}^*$. Since $I_i^D \models C_i$, $\exists r.D_2 \notin I_i^D$. This implies that there is a clause C_k with $i < k < j$, such that the maximal literal in C_k is $\exists r.D_2$. But due to the constraints on the ordering $\prec_l$, $\exists r.D_2 \prec_l \forall r.D_1$, and therefore $C_k \prec C_i$ and $k < i$. We derived a contradiction starting from the assumption $I_j^D \not\models C_i$, and therefore $I_j^D \models C_i$.

implies $I_j^D \models C$ for all $j > i$. □

Next we prove the other part of monotonicity, that is, if the model construction fails to make a clause C_i satisfied by I_i^D, then C_i is also not satisfied by all subsequent model fragments I_j^D with $j > i$. We formulate this lemma stronger than the other monotonicity lemma, since this will be useful later on.

Lemma 4.2.14. *Assume $C \prec_c C_i$ or $C = C_i$, and $I_i^D \not\models C$. Then, $I_j^D \not\models C$ for all $j > i$.*

Proof. Suppose there is a clause $C \prec_c C_i$ with $I_i^D \not\models C$. We do the proof by contradiction and assume that there is a $j > i$ with $I_j^D \models C$. Then, there is a literal $L \in C$ such that $I_i^D \not\models L$ and $I_j^D \models L$.

We distinguish the different cases for L. Observe that since the model construction only adds literals for subsequent model fragments, we have $I_i^D \subseteq I_j^D$.

1. L cannot be of the form $\neg A$, since then we would have $A \in I_i^D$ and $A \notin I_j^D$, which contradicts $I_i^D \subseteq I_j^D$.

2. L cannot be of the form $\forall r.D$, since then there would be a literal $\exists r.D_2$ with $D_2 \neq D_1$ and $\neg D_2 \sqcup D_2 \in \mathcal{N}^*$ such that $\exists r.D_2 \in I_i^D$ and $\exists r.D_2 \notin I_j^D$, which contradicts $I_i^D \subseteq I_j^D$.

3. Assume L is of the form A. Then, $A \notin I_i^D$ and $A \in I_j^D$. This implies that there is a clause C_k with $i < k < j$ in which A is maximal. This is not possible, since with our ordering this clause would be smaller or equal to C and C_i.

4. Assume L is of the form $\exists r.D_1$. Then, there must be a clause $\exists r.D_2 \in (I_j^D \setminus I_i^D)$ such that either $D_1 = D_2$ or $\neg D_2 \sqcup D_1 \in \mathcal{N}^*$. $\exists r.D_2 \in (I_j^D \setminus I_i^D)$ implies that there is a clause C_k with $i < k < j$, such that $\exists r.D_2$ is maximal. If $D_1 = D_2$, then $\exists r.D_1 = \exists r.D_2$. If $\neg D_2 \sqcup D_1 \in \mathcal{N}^*$, then $D_2 \prec_d D_1$ and $\exists r.D_2 \prec_l \exists r.D_1$. In both cases, since $\exists r.D_2$ is maximal in C_k, we obtain that either $C_k = C$ or $C_k \prec_l C$, and also $C_k = C_i$ or $C_k \prec_l C_i$, which contradicts $i < k$. Hence, there can be no literal $\exists r.D_2 \in (I_j^D \setminus I_i^D)$ such that either $D_1 = D_2$ or $\neg D_2 \sqcup D_1 \in \mathcal{N}^*$. Therefore, $I_j^D \not\models C$.

We have shown that C cannot contain any literal L such that $I_i^D \not\models L$ and $I_j^D \models L$. Hence, there cannot be any clause $C \prec C_i$ such that $I_i^D \not\models C$ and $I_j^D \models C$. □

Now we have everything we need, and can show that the construction of the model fragments is successful for each satisfiable definer. That is, we prove that each model fragment satisfies all clauses in the clause set, unless it corresponds to an unsatisfiable definer.

Lemma 4.2.15. *Let D be a definer with $\neg D \notin \mathcal{N}^*$ or $D = \epsilon$. Then, $I^D \models C$ for all $C \in \mathcal{N}^*$.*

Proof. The proof is by contradiction. Assume C_i is the smallest clause according to $\prec_c$ for which $I^D \not\models C_i$, and assume C_i is of the form $C_i' \sqcup L$, where L is the maximal literal in C_i. Since $I^D \not\models C_i$, also $I^D \not\models L$. We distinguish the cases for L.

1. L is of the form A or $\exists r.D'$. Then $I^D \models L$ follows directly from the model construction and Lemma 4.2.13.

2. L is of the form $\neg A$, where $A \neq D$. $I^D \not\models \neg A$ implies $A \in I^D$. If $A \in I^D$ and $A \neq D$, there must be a clause $C_j = C_j' \sqcup A$ in which A is maximal and for which

$I^D_{j-1} \not\models C_j$. Since $C'_j \prec_c C_j$, $I^D \not\models C'_j$ by Lemma 4.2.14. Due to the resolution rule, there then is also a clause $C_k = C'_i \sqcup C'_j$ that is the resolvent of C_i and C_j. $\neg A$ is maximal in C_i, and A is maximal in C_j. Therefore, $\neg A$ is larger than all literals in C_k, and $C_k \prec_c C_i$. We have both $I^D \not\models C'_i$ and $I^D \not\models C'_j$, which implies $I^D \not\models C'_i \sqcup C'_j = C_k$. But then, there is a clause C_k that is smaller than C_i and for which $I^D \not\models C_k$, which contradicts the initial assumption that C_i is the smallest clause for which $I^D \not\models C_i$.

3. L is of the form $\neg D$. Since $\neg D \notin \mathcal{N}^*$ by assumption of the lemma, I^D_0 is initialised with D, and I^D contains D by definition. Therefore, $\neg D$ is not satisfied by I^D regardless of the other clauses in $\mathcal{N}^*$. Since $\neg D \notin \mathcal{N}^*$, C'_i is non-empty. Moreover, since $\neg D$ is maximal and due to how $\prec_l$ is defined, all literals in C'_i are definer literals. Also, they are all negative definer literals, since otherwise, by Lemma 4.2.5, C_i would be of the form $\neg D \sqcup D'$, which implies $D \prec_l D'$, but $\neg D$ is maximal in C_i.

   Let $\neg D_1$ be the maximal literal in C'_i, and let $C'_i = C''_i \sqcup \neg D_1$. $I^D \not\models C_i$ implies $D_1 \in I^D$, which means there is a clause $C_j = C'_j \sqcup D_1$ in which D_1 is maximal, and for which $I^D_{j-1} \not\models C_j$. Resolution on D_1 results in a clause $C_k = \neg D \sqcup C''_i \sqcup C'_j$. $C_k \prec_c C_i$, since $\neg D_1$ is larger than all literals in $C''_i \sqcup C'_j$. Furthermore, $I^D \not\models C_k$, since $I^D \not\models C'_j$ by Lemma 4.2.13 and $I^D \not\models \neg D \sqcup C''_i$ by assumption. But this contradicts the initial assumption that C_i is the smallest clause that is not satisfied by I^D.

4. L is of the form $\forall r.D_1$. $I^D \not\models L$ implies that there is a literal $\exists r.D_2 \in I^D$ with $D_1 \neq D_2$ and $\neg D_2 \sqcup D_1 \notin \mathcal{N}^*$. If $\exists r.D_2 \in I^D$, there is a clause $C_j = C'_j \sqcup \exists r.D_2$ in which $\exists r.D_2$ is maximal and for which $I^D_{j-1} \not\models C_j$. Together with Lemma 4.2.14, this implies $I^D \not\models C'_j$. Due to the role propagation rule, there is also a clause $C_k = C'_i \sqcup C'_j \sqcup \exists r.D_{12}$, where D_{12} represents $D_1 \sqcap D_2$. The latter implies that there are the clauses $\neg D_{12} \sqcup D_1, \neg D_{12} \sqcup D_2 \in \mathcal{N}^*$. $C_j \prec_c C_i$, since for the maximal literals we have $\exists r.D_2 \prec_l \forall r.D_1$. Since $\exists r.D_{12} \prec_c \exists r.D_2$, also $C_k \prec C_j$. Note that $I_{j-1} \not\models \exists r.D_{12}$, since otherwise we would not have have $I_{j-1} \not\models C_j$. By Lemma 4.2.14, $I \not\models \exists r.D_{12}$. But then, because also $I \not\models C'_i \sqcup C'_j$, $I \not\models C_k$. This contradicts our initial assumption that C^D_i is the smallest clause that is not

satisfied by I^D.

In all four cases we have a contradiction, and therefore, there cannot be a smallest clause C_i with $I^D \not\models C_i$. Hence, $I^D \models C$ for all clauses $C \in \mathcal{N}^*$. □

Before we continue defining the actual candidate model, we note some further properties of the generated model fragments, since these are used in later proofs.

Lemma 4.2.16. *If $\neg D \in \mathcal{N}^*$, then no model fragment for $\mathcal{N}^*$ contains a literal of the form $\exists r.D$.*

Proof. Assume that $\neg D \in \mathcal{N}^*$ and that there is a model fragment I^{D_1} such that $\exists r.D \in I^{D_1}$. Literals of the form $\exists r.D$ are only added to a model fragment I^{D_1} if there exists a clause $C_i = C_i' \sqcup \exists r.D$, where $\exists r.D$ is maximal, such that $I_{i-1}^{D_1} \not\models C_i$. By Lemma 4.2.14, we also have $I^{D_1} \not\models C_i'$ in this case. Since $C_i' \sqcup \exists r.D \in \mathcal{N}^*$ and $\neg D \in \mathcal{N}^*$, the $\exists$-elimination rule applies, and there is also a clause $C_i' \in \mathcal{N}^*$. By Lemma 4.2.15, $I^D \models C_i'$. But this contradicts the earlier observation that $I^D \not\models C_i'$. Hence, our initial assumption that $\exists r.D \in I^{D_1}$ cannot be true, and no model fragment for $\mathcal{N}^*$ contains a literal of the form $\exists r.D$ if $\neg D \in \mathcal{N}^*$. □

Lemma 4.2.17. *If $D_2 \in I^{D_1}$, where $D_2 \in N_d$, then either $D_2 = D_1$, or there is a clause $\neg D_1 \sqcup D_2 \in \mathcal{N}^*$.*

Proof. By Lemma 4.2.5, positive definer literals only occur in clauses of the form $\neg D \sqcup D'$. By induction on the model construction, and the fact that $I_0^{D_1} = \{D_1\}$, we can show that there is a sequence of clauses of the form $\neg D_1 \sqcup D_1'$, $\neg D_1' \sqcup D_2'$, …, $\neg D_n' \sqcup D_2$ if $D_2 \in I^{D_1}$. By applying pair-wise resolution on these clauses we infer the clause $\neg D_1 \sqcup D_2$, which is included in $\mathcal{N}^*$. □

Observe that an additional consequence of this lemma is that D is always the smallest definer literal in I^D.

Based on the model fragments, we build a candidate model $\mathcal{I}$, and show that it is indeed a model of $\mathcal{N}^*$. $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ is defined as follows:

1. $\Delta^{\mathcal{I}} = \{x_D \mid D \in N_d \cup \{\epsilon\}, \neg D \notin \mathcal{N}^*\}$
2. For every $A \in N_c$, $A^{\mathcal{I}} = \{x_D \mid A \in I^D\}$

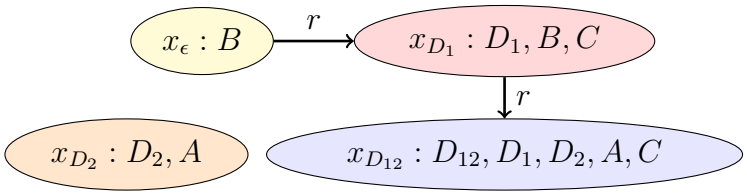

Figure 4.7: The candidate model $\mathcal{I}_3$ built throughout Examples 4.2.8 and 4.2.18.

3. For every $r \in N_r$, $r^{\mathcal{I}} = \{(x_{D_1}, x_{D_2}) \mid \exists r.D_2 \in I^{D_1}\}$

For every definer $D \in \mathcal{N}^*$ with $\neg D \notin \mathcal{N}^*$, the candidate model contains a corresponding domain element x_D. Observe that for the remaining definers D, $I^D = \emptyset$. Therefore, they are not involved in the interpretations of concepts and roles, which are defined following the elements of the model fragments. For the interpretation function for roles, observe that by Lemma 4.2.16, we only have literals of the form $\exists r.D_2 \in I^{D_1}$ if $\neg D_2 \notin \mathcal{N}^*$. Hence, if $\exists r.D_2 \in I^{D_1}$, there is always a domain element $x_{D_2} \in \Delta^{\mathcal{I}}$.

Example 4.2.18. For the model fragments we created in Example 4.2.8, the candidate model $\mathcal{I}_3 = \langle \Delta^{\mathcal{I}_3}, \cdot^{\mathcal{I}_3} \rangle$ has the following form:

$$\Delta^{\mathcal{I}_3} = \{x_\epsilon, x_{D_1}, x_{D_2}, x_{D_{12}}\}$$
$$A^{\mathcal{I}_3} = \{x_{D_2}, x_{D_{12}}\}$$
$$B^{\mathcal{I}_3} = \{x_\epsilon, x_{D_1}\}$$
$$C^{\mathcal{I}_3} = \{x_{D_1}, x_{D_{12}}\}$$
$$D_1^{\mathcal{I}_3} = \{x_{D_1}, x_{D_{12}}\}$$
$$D_2^{\mathcal{I}_3} = \{x_{D_2}, x_{D_{12}}\}$$
$$D_{12}^{\mathcal{I}_3} = \{x_{D_{12}}\}$$
$$r^{\mathcal{I}_3} = \{(x_\epsilon, x_{D_1}), (x_{D_1}, x_{D_{12}})\}$$

A visualisation of this candidate model is shown in Figure 4.7. Let us recall the ontology $\mathcal{O}_3$ from which the example started:

$$\top \sqsubseteq A \sqcup B$$
$$B \sqsubseteq \exists r.C$$
$$C \sqsubseteq \forall r.A$$

It can easily be verified that the axioms of the ontology are satisfied by all elements of the candidate model, and that $\mathcal{I}_3$ is a model of $\mathcal{O}_3$.

We prove that the model construction not only succeeds in this particular example, but for every set of clauses $\mathcal{N}^*$ saturated using $Res_{\mathcal{ALC}}$.

Lemma 4.2.19. *$\mathcal{I}$ is a model for $\mathcal{N}^*$.*

Proof. The proof is by contradiction. Assume C_i is the smallest clause in $\mathcal{N}^*$ that is not satisfied by $\mathcal{I}$. Then, there is a domain element x_D such that $x_D \notin (C_i)^{\mathcal{I}}$. By Lemma 4.2.15, $I^D \models C_i$. Therefore, C_i contains a literal L such that $I^D \models L$. Since $x_D \notin (C_i)^{\mathcal{I}}$, $x_D \notin L^{\mathcal{I}}$. We distinguish the different cases for L, and show that each case contradicts $x_D \notin L^{\mathcal{I}}$.

1. Assume $L = A$. $I^D \models A$ implies $A \in I^D$. Hence, $x_D \in A^{\mathcal{I}}$ follows immediately from the construction of the candidate model, which contradicts $x_D \notin L^{\mathcal{I}}$.

2. Assume $L = \neg A$. $I^D \models \neg A$ implies $A \notin I^D$. Again, $x_D \notin A^{\mathcal{I}}$ follows immediately from the model construction, which contradicts $x_D \notin L^{\mathcal{I}}$.

3. Assume $L = \exists r.D_1$. $I^D \models \exists r.D_1$ implies $\exists r.D_2 \in I^D$ with either $D_2 = D_1$ or $\neg D_2 \sqcup D_1 \in \mathcal{N}^*$. Additionally, by Lemma 4.2.16, $\neg D_2 \notin \mathcal{N}^*$, which ensures $x_{D_2} \in \Delta^{\mathcal{I}}$. Therefore, by model construction, $(x_D, x_{D_2}) \in r^{\mathcal{I}}$. By construction of I^{D_2}, also $D_2 \in I^{D_2}$, and hence $x_{D_2} \in D_2^{\mathcal{I}}$. If $D_2 = D_1$, this directly implies $x_D \in (\exists r.D_1)^{\mathcal{I}}$. If $D_2 \neq D_1$, $\neg D_2 \sqcup D_1 \in \mathcal{N}^*$. According to the ordering, $\neg D_2 \sqcup D_1 \prec_c C_i$, and hence, due to the initial assumption, $\mathcal{I} \models \neg D_2 \sqcup D_1$, and also $x_{D_2} \in (\neg D_2 \sqcup D_1)^{\mathcal{I}}$. Together with $x_{D_2} \in D_2^{\mathcal{I}}$, this implies $x_{D_2} \in D_1^{\mathcal{I}}$. But then $x_D \in (\exists r.D_1)^{\mathcal{I}}$ is satisfied, which contradicts $x_D \notin L^{\mathcal{I}}$.

4. Assume $L = \forall r.D_1$. $x_D \notin (\forall r.D_1)^{\mathcal{I}}$ implies that there is an edge $(x_D, x_{D_2}) \in r^{\mathcal{I}}$ with $x_{D_2} \notin D_1^{\mathcal{I}}$. The latter additionally implies that $D_1 \notin I^{D_2}$. With the model construction, we only have $(x_D, x_{D_2}) \in r^{\mathcal{I}}$ if $\exists r.D_2 \in I^D$. By construction of $I_0^{D_2}$, we further have $D_2 \in I^{D_2}$. Since $I^D \models \forall r.D_1$, either $D_2 = D_1$ or $\neg D_2 \sqcup D_1 \in \mathcal{N}^*$. $D_2 \neq D_1$, since otherwise $D_1 \in I^{D_2}$. But then, since $D_2 \in I^{D_2}$ and $D_1 \notin I^{D_2}$, $x_{D_2} \notin (\neg D_2 \sqcup D_1)^{\mathcal{I}}$, which contradicts the initial assumption that C_i is the smallest clause not satisfied by $\mathcal{I}$.

Each case leads to a contradiction, which means that there cannot be a domain element $x_D \in \Delta^{\mathcal{I}}$ such that $x_D \notin (C_i)^{\mathcal{I}}$. This contradicts the initial assumption that there is a smallest clause C_i that is not satisfied by $\mathcal{I}$. We obtain that $\mathcal{I}$ is a model of $\mathcal{N}^*$. □

Lemma 4.2.19 allows us to establish that $Res_{\mathcal{ALC}}$ is a sound and refutationally complete decision procedure for $\mathcal{ALC}\nu$ ontology satisfiability.

Theorem 4.2.20. *$Res_{\mathcal{ALC}}$ is sound, terminating and refutationally complete, and provides a decision procedure for $\mathcal{ALC}\nu$ ontology satisfiability.*

Proof. By Theorem 4.1.2, any $\mathcal{ALC}\nu$ ontology $\mathcal{O}$ can be transformed into the equisatisfiable clause set $Cl(\mathcal{O})$ that can be processed by $Res_{\mathcal{ALC}}$. Lemma 4.2.3 establishes soundness of $Res_{\mathcal{ALC}}$ and Lemma 4.2.4 termination. Finally, Lemma 4.2.19 establishes refutational completeness by showing how a model can be constructed from any saturated set of clauses that does not contain the empty clause. □

4.3 Refinements and Redundancy Elimination

Before we extend $Res_{\mathcal{ALC}}$ to a calculus for computing uniform interpolants, we introduce refinements and redundancy elimination for $Res_{\mathcal{ALC}}$ as a decision procedure. The refined calculus is used as basis for the uniform interpolation method, and for the proofs of its correctness. Furthermore, redundancy elimination is necessary in order to make the calculus practical, which is of particular importance since the overall aim is to develop a practical method for computing uniform interpolants based on this calculus.

If we look at the clauses used in Examples 4.2.8 to 4.2.18 to illustrate the model construction (Figure 4.6), we observe that the majority of these clauses does not contribute to the constructed model. As noted in the beginning of the chapter, we want to avoid inferences that are not required, in order to obtain a practical method. Another observation from the examples is that the calculus derives a lot of clauses that contain more than one negative definer literal. The technique for eliminating definer symbols presented in Section 4.1.2 can only deal with clauses that contain at most one negative definer literal. If the calculus is to be used for uniform interpolation, we have

to be able to eliminate all definers that are introduced by the method. As it turns out, this is not a problem in practice, since inferences of clauses with more than one negative definer can be omitted. Recall from Definition 4.1.4 that a clause is normal if it contains at most one negative definer literal.

Definition 4.3.1. Given a calculus Calc, Calc^{norm} refers to a refinement of Calc such that each rule of the calculus is only applied if the conclusion contains at most one negative definer literal.

Lemma 4.3.2. *Let $\mathcal{N}$ be a set of normal clauses, $\mathcal{N}^* \subseteq Res_{\mathcal{ALC}}(\mathcal{N})$ be some subset of the saturation of $\mathcal{N}$ using our calculus, and $C_i = \neg D_1 \sqcup \neg D_2 \sqcup C_i \in \mathcal{N}^*$ be a clause with more than one negative definer literal. Then, we obtain the same model fragments if we base the construction on the clause set $\mathcal{N}^* \setminus \{C_i\}$ as we do if we base the construction on $\mathcal{N}^*$.*

Proof. Since the normal form transformation does not introduce clauses with more than one negative definer literal, there are only two possibilities why C_i is in the clause set: (1) C_i is the conclusion of a rule application on two clauses that each have a different negative definer literal or (2) C_i is the conclusion of a rule application on at least one clause that has more than one negative definer. If we can establish that inferences as in Case 1 are not necessary, we do not have to consider Case 2, since it is only possible if we perform inferences as in Case 1. We show that inferences as in Case 1 indeed have no impact on the model generation.

Assume C_i is the result of applying a rule of the calculus on two clauses $C_{j_1} = \neg D_1 \sqcup C'_{j_1}$ and $C_{j_2} = \neg D_2 \sqcup C'_{j_2}$. The model construction only adds literals to a model fragment I^D if $I^D_{i-1} \not\models C_i$. If C_i has an effect on the model fragments, there must therefore be a definer D with $I^D_{i-1} \not\models C_i$. This implies $I^D_{i-1} \not\models \neg D_1 \sqcup \neg D_2$, and therefore $D_1, D_2 \in I^D_{i-1}$. By Lemma 4.2.17, we then have for each $a \in \{1, 2\}$ either $D_a = D$ or $\neg D \sqcup D_a \in \mathcal{N}^*$. By resolution on D_1 or D_2 or both, we obtain that then, there are also the clauses $C_{j'_1} = \neg D \sqcup C'_{j_1} \in \mathcal{N}^*$ and $C_{j'_2} = \neg D \sqcup C'_{j_2} \in \mathcal{N}^*$. The same rule with which C_i is inferred from C_{j_1} and C_{j_2} can be used to infer the clause $C_{i'} = \neg D \sqcup C'$ from $C_{j'_1}$ and $C_{j'_2}$. Since either $D_a = D$ or $\neg D \sqcup D_a \in \mathcal{N}^*$ for D_a, $a \in \{1, 2\}$, either $D_a = D$ or $D \prec_d D_a$. Therefore, $C_{i'} \prec_c C_i$. Due to Lemma 4.2.15, $I^D_{i'} \models C_{i'}$, and by monotonicity (Lemma 4.2.13), $I^D_{i-1} \models C_i$. Therefore, C_i has no

influence on the constructed model fragments, and if we remove C_i from the clause set, we still obtain the same model. □

Theorem 4.3.3. *$Res^{norm}_{\mathcal{ALC}}$ is sound and refutationally complete, and provides a decision procedure for $\mathcal{ALC}\nu$ ontology satisfiability.*

Proof. By Lemma 4.3.2, for every clause C with more than one negative definer literal and for every model fragment I^D, $I^D \models C$. Hence, if we remove these clauses, the model construction is still successful. □

The next refinements add ordered resolution and redundancy elimination to the calculus, in a similar way as it is done in Bachmair and Ganzinger (2001), and is used in many popular resolution-based theorem provers.

We first refine $Res_{\mathcal{ALC}}$ using the ordering we defined in Section 4.2.3.

Definition 4.3.4. Given a calculus Calc and an ordering $\prec_l$ on literals, $\mathsf{Calc}^{\prec_l}$ refers to a refinement of Calc, in which all the rules of the calculus are only applied on definer literals or on literals which are maximal according to $\prec_l$.

Inspection of all proofs needed to establish Lemma 4.2.15 for refutational completeness reveals that we only refer to rule applications on maximal literals in a clause. Other rule inferences are therefore not required for refutational completeness, if no other restrictions are applied on the calculus. The only proof which relies on inferences on symbols which are not maximal is the proof for Lemma 4.3.2, which is required to establish completeness of $Res^{norm}_{\mathcal{ALC}}$. This proof relies on applications of the resolution rule on definer literals. These inferences are the only exceptions in $\mathsf{Calc}^{\prec_l}$ where the ordering $\prec_l$ can be ignored. Note that the only possible inference where both literals are definer literals are resolution inferences. By Lemma 4.2.5, positive definer literals only occur in clauses of the form $\neg D \sqcup D$. Intuitively, ignoring the ordering for inferences on definer literals serves the purpose of supporting the mechanism for introducing definers.

With the above observations, we can establish the following theorem.

Theorem 4.3.5. *Let $\prec_l$ be any ordering on literals that satisfies the Constraints 1–6 on page 65. Then, $Res^{norm^{\prec_l}}_{\mathcal{ALC}}$ is sound and refutationally complete, and provides a decision procedure for $\mathcal{ALC}\nu$ ontology satisfiability.*

As in traditional resolution-based decision procedures, it is possible to extend the method with redundancy elimination and further simplification techniques. For our notion of redundancy, we make use of the relations between introduced definers. Note that new definers are introduced by adding clauses of the form $\neg D_1 \sqcup D_2$. The clause $\neg D_1 \sqcup D_2$ is equivalent to the concept inclusion $D_1 \sqsubseteq D_2$. This concept inclusion can be used to define subsumption between existential and universal role restrictions, as well as subsumption between clauses.

Definition 4.3.6. A literal L_1 *subsumes a literal* L_2, in symbols $L_1 \sqsubseteq_l L_2$, if either $L_1 = L_2$, or if $L_1 = \mathsf{Q}r.D_1$ and $L_2 = \mathsf{Q}r.D_2$ for $\mathsf{Q} \in \{\exists, \forall\}$, and there is a clause $\neg D_1 \sqcup D_2$ in the current clause set. A clause C_1 *subsumes a clause* C_2, in symbols $C_1 \sqsubseteq_c C_2$, if every literal $L_1 \in C_1$ subsumes a literal $L_2 \in C_2$. A clause C is *redundant* with respect to a clause set $\mathcal{N}$, if $\mathcal{N}$ contains a clause C' with $C' \sqsubseteq_c C$. The *reduction* of a clause C, which we denote by *red*(C), is obtained from C by removing every literal that subsumes another literal in C.

The notion of subsumption as it is defined here follows the tradition used in the literature on theorem proving and resolution. Note that while a clause is redundant in a set of clauses if it is *subsumed* by another clause in $\mathcal{N}$, a literal is redundant in a clause C if it *subsumes* another literal in C. The reason is that a set of clauses is essentially a conjunction of clauses, whereas a clause is a disjunction of literals. For this reason, the reduction of a clause deletes *subsuming* literals, whereas subsumption deletion in a clause set deletes *subsumed* clauses.

Example 4.3.7 (Subsumption and reduction)**.** Assume D_{12} represents $D_1 \sqcap D_2$, which means we have the clauses $\neg D_{12} \sqcup D_1$ and $\neg D_{12} \sqcup D_2$. Then, the following relations hold:

$$\neg A \sqcup B \sqsubseteq_c \neg A \sqcup B \sqcup C,$$

$$\exists r.D_{12} \sqsubseteq_l \exists r.D_1,$$

$$\forall r.D_{12} \sqcup B \sqsubseteq_c \forall r.D_1 \sqcup A \sqcup B,$$

$$\mathit{red}(A \sqcup \exists r.D_{12} \sqcup \exists r.D_2) = A \sqcup \exists r.D_2.$$

In addition to subsumption and reduction, we also detect tautological clauses which contain pairs of contradictory literals. This leads to the set of simplification rules

Tautology Deletion:	$\dfrac{N \cup \{C \sqcup A \sqcup \neg A\}}{N}$	
Subsumption Deletion:	$\dfrac{N \cup \{C,\ D\}}{N \cup \{C\}}$	provided $C \sqsubseteq_c D$
Reduction:	$\dfrac{N \cup \{C\}}{N \cup \{\mathit{red}(C)\}}$	

Figure 4.8: Simplification rules.

shown in Figure 4.8. We denote the calculus $Res_{\mathcal{ALC}}^{norm^{\prec_l}}$ extended with these rules by $Res_{\mathcal{ALC}}^{norm^{\prec_l}s}$. It can be shown that the rules preserve soundness and refutational completeness, as stated by the following theorem.

Theorem 4.3.8. *$Res_{\mathcal{ALC}}^{norm^{\prec_l}s}$ is sound and refutationally complete and provides a decision procedure for $\mathcal{ALC}\nu$ ontology satisfiability.*

Proof. Since the tautology deletion and the subsumption deletion rule only remove clauses, they do not affect the soundness of the calculus. The soundness of the reduction rule follows from the facts that 1) if $L_1 \sqsubseteq_l L_2$ holds in $\mathcal{N}$, then $\mathcal{N} \models L_1 \sqsubseteq L_2$, and 2) if $\mathcal{N} \models L_1 \sqsubseteq L_2$, then $\mathcal{N} \models L_1 \sqcup L_2 \sqsubseteq L_2$, in which case $\mathcal{N} \cup \{C_1 \sqcup L_1 \sqcup L_2\} \models C_1 \sqcup L_2$.

We show that the refutational completeness is preserved by each rule by referring to the candidate model construction in Lemma 4.2.15.

Tautology deletion: Assume $C = C' \sqcup A \sqcup \neg A$. Since both $A \in C$ and $\neg A \in C$, for any literal set I^D we have $I^D \models C$. Therefore, the presence of C does not affect the candidate model construction, and it can as well be removed from the clause set.

Subsumption deletion: Observe that $L_1 \sqsubseteq_l L_2$ implies either $L_1 = L_2$ or $L_1 \prec_l L_2$, where $\prec_l$ is the ordering used in the model construction for Theorem 4.3.8. Hence, $C_1 \sqsubseteq_c C_2$ implies either $C_1 = C_2$ or $C_1 \prec_c C_2$. Observe also that due to Definition 4.2.11, for any model fragment I, if $C_1 \sqsubseteq_c C_2$, $I \models C_1$ implies $I \models C_2$. Therefore, C_2 does not affect the candidate model construction and can as well be removed from the clause set.

Reduction: Since $\mathit{red}(C) \sqsubseteq_c C$ and subsumption deletion preserves refutational completeness, reduction preserves refutational completeness as well. □

4.4 The Interpolation Procedure

4.4.1 Extending the Calculus

The calculus $Res_{\mathcal{ALC}}$ is sound, refutationally complete and terminating. By Theorem 4.3.3, refutational completeness is preserved if only clauses with at most one negative definer literal are inferred. This allows us to restrict the calculus in such a way that every saturated set of clauses can be translated back into an ontology without definer symbols, following Theorem 4.1.5. Moreover, by Theorem 4.3.5, if we are given a signature $\mathcal{S}$, the calculus can be restricted using an ordering in such a way that inferences on symbols which are not in $\mathcal{S}$ are performed before other inferences.

In order to compute a uniform interpolant, we want to use the calculus to saturate the normal form representation of an input ontology until clauses outside of the desired signature, modulo definers, can be removed. Elimination of the introduced definer symbols should then result in the uniform interpolant for that signature. Whereas the above features are desirable for using a calculus for this purpose, they are not sufficient, since they are solely based on the notion of refutational completeness. Refutational completeness is not sufficient for our purposes, since the only entailment it guarantees to be derivable is the empty clause. To express the appropriateness of a calculus for uniform interpolation, we need a stronger notion of completeness.

For any set of clauses $\mathcal{N}$ and signature $\mathcal{S}$, denote by $\mathcal{N}_\mathcal{S}$ the subset of clauses C in $\mathcal{N}$ that are in $\mathcal{S}$ modulo definers, that is $\mathcal{N}_\mathcal{S} = \{C \mid sig(C) \subseteq \mathcal{S} \cup N_d\}$. Recall from Definition 4.1.4 that $\mathcal{N}_n$ denotes the subset of normal clauses in $\mathcal{N}$, where a normal clause is a clause with at most one negative definer symbol. *Cl* and *Ont* where defined in Sections 4.1 and 4.1.2 as functions transforming $\mathcal{ALC}\nu$ ontologies without definers into normal clause sets, and normal clause sets into $\mathcal{ALC}\nu$ ontologies without definers.

Definition 4.4.1. A calculus Calc is *interpolation complete* for a language $\mathcal{L}$, if for any $\mathcal{L}$ ontology $\mathcal{O}$ and signature $\mathcal{S}$, $Ont((\mathcal{N}_n^*)_\mathcal{S})$ is an $\mathcal{L}$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$, where $\mathcal{N}^* = \mathsf{Calc}(Cl(\mathcal{O}))$.

Theorem 4.4.2. *$Res_{\mathcal{ALC}}$ is not interpolation complete for $\mathcal{ALC}$.*

$\forall\forall$-Role Propagation

$$\frac{C_1 \sqcup \forall r.D_1 \qquad C_2 \sqcup \forall r.D_2}{C_1 \sqcup C_2 \sqcup \forall r.D_{12}}$$

where D_{12} is a possibly new definer representing $D_1 \sqcap D_2$.

Figure 4.9: Additional rule for the interpolation complete calculus $Int_{\mathcal{ALC}}$.

Proof. Consider the following ontology $\mathcal{O}_4$:

$$A \sqsubseteq \forall r.B$$

$$A \sqsubseteq \forall r.(\neg B \sqcup C)$$

One can verify that $\mathcal{O} \models A \sqsubseteq \forall r.C$.

$\mathcal{N}_4^* = Res_{\mathcal{ALC}}(\mathcal{N}_4)$, where $\mathcal{N}_4 = Cl(\mathcal{O}_4))$, contains the following clauses:

1. $\neg A \sqcup \forall r.D_1$	*(Normal Form Transformation)*
2. $\neg D_1 \sqcup B$	*(Normal Form Transformation)*
3. $\neg A \sqcup \forall r.D_2$	*(Normal Form Transformation)*
4. $\neg D_2 \sqcup \neg B \sqcup C$	*(Normal Form Transformation)*
5. $\neg D_1 \sqcup \neg D_2 \sqcup C$	*(Resolution 2, 4)*

Let $\mathcal{S} = \{A, r, C\}$. Observe that the only clause inferred by $Res_{\mathcal{ALC}}$, Clause 5, has two negative definer literals. Therefore, $(\mathcal{N}_4^*)_n$ is equal to the initial clause set $\mathcal{N}_4$. Consequently, $\mathcal{N}_4^{\mathcal{S}} = ((\mathcal{N}_4^*)_n)_{\mathcal{S}}$ contains only Clause 1 and 3, which are marked with a bold face number. D_1 and D_2 occur only positively in $\mathcal{N}_4^{\mathcal{S}}$, and are therefore replaced by $\top$ when $Ont(\mathcal{N}_4^{\mathcal{S}})$ is computed (see Section 4.1.2). $\mathcal{O}_4' = Ont(\mathcal{N}_4^{\mathcal{S}})$ contains therefore only one axiom, which is in fact a tautology:

$$\top \sqsubseteq \neg A \sqcup \forall r.\top$$

Clearly, $\mathcal{O}_4' \not\models A \sqsubseteq \forall r.C$, even though $\mathcal{O}_4 \models A \sqsubseteq \forall r.C$. Hence, $\mathcal{O}_4'$ is not an $\mathcal{ALC}$ uniform interpolant of $\mathcal{O}_4$ for $\mathcal{S}$. Therefore, $Res_{\mathcal{ALC}}$ is not interpolation complete. □

We need an additional rule for handling interactions between universal restrictions. This rule, shown in Figure 4.9, is sufficient to transform $Res_{\mathcal{ALC}}$ into an interpolation complete calculus for $\mathcal{ALC}$.

The $\forall\forall$-role propagation rule is similar to the $\forall\exists$-role propagation rule. Whereas the $\forall\exists$-role propagation rule propagates universally quantified information into existential role restrictions, the $\forall\forall$-role propagation rule propagates universally quantified information into universal role restrictions. Both rules may lead to the introduction of new definer symbols. The soundness of the $\forall\forall$-role propagation rule follows from a similar observation as for the $\forall\exists$-role propagation rule, namely from the valid entailment $\models (\forall r.D_1 \sqcap \forall r.D_2) \sqsubseteq (\forall r.(D_1 \sqcap D_2))$.

$Int_{\mathcal{ALC}}$ extends $Res^{norm}_{\mathcal{ALC}}$ with the $\forall\forall$-role propagation rule. Note that $Res^{norm}_{\mathcal{ALC}}$ only allows inferences with at most one negative definer literal. By Theorem 4.3.3, and since we only add sound inferences to the calculus, $Int_{\mathcal{ALC}}$ is still sound and refutationally complete. $Int_{\mathcal{ALC}}$ can be extended with redundancy elimination and ordered resolution, but for current purposes this is not necessary. How redundancy elimination is used for the computation of uniform interpolants is discussed later in this section.

The example used to prove Theorem 4.4.2 can be solved correctly using $Int_{\mathcal{ALC}}$.

Example 4.4.3. We again consider the ontology $\mathcal{O}_4$:

$$A \sqsubseteq \forall r.B$$
$$A \sqsubseteq \forall r.(\neg B \sqcup C)$$

$\mathcal{N}_4 = Cl(\mathcal{O}_4)$ consists of the following clauses:

1. $\neg A \sqcup \forall r.D_1$	*(Normal Form Transformation)*
2. $\neg D_1 \sqcup B$	*(Normal Form Transformation)*
3. $\neg A \sqcup \forall r.D_2$	*(Normal Form Transformation)*
4. $\neg D_2 \sqcup \neg B \sqcup C$	*(Normal Form Transformation)*

Using $Int_{\mathcal{ALC}}$, we cannot apply resolution on Clause 2 and Clause 4, because the resulting clause would contain more than one negative definer literal. We can however apply the $\forall\forall$-role propagation rule on Clause 1 and Clause 3, with the effect that a new definer for $D_1 \sqcap D_2$ is introduced. This makes further resolution steps possible.

5. $\neg A \sqcup \forall r.D_{12}$	*($\forall\forall$-role propagation 1, 3)*
6. $\neg D_{12} \sqcup D_1$	*($D_{12} \sqsubseteq D_1$)*
7. $\neg D_{12} \sqcup D_2$	*($D_{12} \sqsubseteq D_2$)*

8. $\neg D_{12} \sqcup B$ *(Resolution 2, 6)*

9. $\neg D_{12} \sqcup \neg B \sqcup C$ *(Resolution 4, 7)*

10. $\neg D_{12} \sqcup C$ *(Resolution 8, 9)*

The set of clauses is saturated with respect to $Int_{\mathcal{ALC}}$. We denote it by $\mathcal{N}_4^*$. Again we set $\mathcal{S} = \{A, r, C\}$. The set $(\mathcal{N}_4^*)_\mathcal{S}$, that is, the clauses whose signatures are fully in $\mathcal{S} \cup N_d$, contains the clauses with a bold face number. Since only normal clauses are derived in $Int_{\mathcal{ALC}}$, $(\mathcal{N}_4^*)_\mathcal{S} = ((\mathcal{N}_4^*)_n)_\mathcal{S}$. Therefore, there are no non-normal clauses to remove. If $Int_{\mathcal{ALC}}$ is interpolation complete, $Ont((\mathcal{N}_4^*)_\mathcal{S})$ should be a uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$. $Ont((\mathcal{N}_4^*)_\mathcal{S})$ contains the following axioms:

$$\top \sqsubseteq \neg A \sqcup \forall r.\top$$
$$\top \sqsubseteq \neg A \sqcup \forall r.(\top \sqcap \top \sqcap C),$$

which can be simplified to the following single axiom:

$$A \sqsubseteq \forall r.C$$

This axiom is as expected. Indeed, $Ont(\mathcal{N}_\mathcal{S}^*)$ is an $\mathcal{ALC}$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$.

4.4.2 Interpolation Completeness of $Int_{\mathcal{ALC}}$

In order to prove that $Int_{\mathcal{ALC}}$ is interpolation complete, we have to show that for any computed result $\mathcal{O}^\mathcal{S}$ of $\mathcal{O}$ for $\mathcal{S}$ and any $\mathcal{ALC}$ axiom α in $\mathcal{S}$, $\mathcal{O}^\mathcal{S} \models \alpha$ iff $\mathcal{O} \models \alpha$. The proof exploits the refutational completeness of $Res_{\mathcal{ALC}}^{norm_{\prec_l}}$, and shows that every proof of $\mathcal{O} \models \alpha$ constructed using $Res_{\mathcal{ALC}}^{norm_{\prec_l}}$ can be transformed into a proof of $\mathcal{O}^\mathcal{S} \models \alpha$ constructed using $Res_{\mathcal{ALC}}^{norm_{\prec_l}}$.

We fix an $\mathcal{ALC}$ ontology $\mathcal{O}$, a signature $\mathcal{S}$ and an $\mathcal{ALC}$ axiom α with $sig(\alpha) \subseteq \mathcal{S}$. Since axioms of the form $C \equiv D$ are logically equivalent to pairs of axioms of the form $C \sqsubseteq D$ and $D \sqsubseteq C$, we can restrict ourselves to the case where α is a concept inclusion of the form $C \sqsubseteq D$. Let $\mathcal{O}^\mathcal{S}$ be an ontology computed using the procedure described in the last section, that is, by saturating the normalised ontology $\mathcal{N} = Cl(\mathcal{O})$ with $Int_{\mathcal{ALC}}$, removing all clauses with symbols outside $\mathcal{S} \cup N_d$, and eliminating all introduced definers using the procedure described in Section 4.1.2. If $Int_{\mathcal{ALC}}$ is interpolation complete, we should have $\mathcal{O} \models C \sqsubseteq D$ iff $\mathcal{O}^\mathcal{S} \models C \sqsubseteq D$.

Note that the ordering $\prec_l$ defined on page 65 is based on a total ordering $\prec_s$ on $(N_c \cup N_r) \setminus N_d$. We can specify $\prec_s$ in such a way that symbols outside $\mathcal{S}$ are larger than symbols in $\mathcal{S}$. This way, we obtain a total ordering $\prec_l$ on literals such that $L_1 \prec_l L_2$ whenever $sig(L_1) \subseteq \mathcal{S} \cup N_d$ and $sig(L_2) \not\subseteq \mathcal{S} \cup N_d$. By Theorem 4.3.5, the calculus $Res_{\mathcal{ALC}}^{norm\prec_l}$ based on this ordering is refutationally complete.

$\mathcal{O} \models C \sqsubseteq D$ can be proved by showing that $\mathcal{O} \cup \{\top \sqsubseteq \exists r^*.(C \sqcap \neg D)\}$ is unsatisfiable, where r^* is fresh. Let $\mathcal{N} = Cl(\mathcal{O})$, $\mathcal{N}^{\mathcal{S}} = Cl(\mathcal{O}^{\mathcal{S}})$ and $\mathcal{M} = Cl(\exists r^*.(C \sqcap \neg D))$, where $r^* \notin sig(\mathcal{O})$. $\mathcal{N} \cup \mathcal{M}$ is unsatisfiable iff $\mathcal{O} \models \alpha$. Interpolation completeness can therefore be shown by proving that $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ is unsatisfiable iff $\mathcal{N} \cup \mathcal{M}$ is unsatisfiable. Because $Res_{\mathcal{ALC}}^{norm\prec_l}$ is refutationally complete, this is equivalent to proving $\bot \in Res_{\mathcal{ALC}}^{norm\prec_l}(\mathcal{N} \cup \mathcal{M})$ iff $\bot \in Res_{\mathcal{ALC}}^{norm\prec_l}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$.

Let $\mathcal{N}^*$ be the saturation of $\mathcal{N}$ by $Int_{\mathcal{ALC}}$, as it is used for computing $\mathcal{N}^{\mathcal{S}}$. For simplicity, we may assume that the same definers are used in $\mathcal{N}^{\mathcal{S}}$ that are used in the saturation of $\mathcal{N}$ for $\mathcal{N}^*$ and $Res_{\mathcal{ALC}}^{norm\prec_l}(\mathcal{N} \cup \mathcal{M})$. Computing $Res_{\mathcal{ALC}}^{norm\prec_l}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$ corresponds to first computing $\mathcal{N}^*$ and then saturating the union of the resulting set and $\mathcal{M}$, where clauses in $\mathcal{N}^*$ with symbols outside $\mathcal{S} \cup N_d$ are not used any more. We have to make sure that all inferences on symbols in $\mathcal{S}$ that are required to infer $\bot$ from $\mathcal{N} \cup \mathcal{M}$ have corresponding inferences in $\mathcal{N}^*$. Some of these inferences may involve definers from $\mathcal{M}$ when $Res_{\mathcal{ALC}}^{norm\prec_l}$ is used. The main challenge is therefore to show that these inferences have corresponding inferences in $\mathcal{N}^*$. For this, we show that every introduced definer in $Res_{\mathcal{ALC}}^{norm\prec_l}(\mathcal{N} \cup \mathcal{M})$ that was necessary for doing these inferences has a corresponding definer in $\mathcal{N}^*$, and that by saturating with $\mathcal{M}$, we obtain similar clauses in $Res_{\mathcal{ALC}}^{norm\prec_l}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$ as in $Res_{\mathcal{ALC}}^{norm\prec_l}(\mathcal{N} \cup \mathcal{M})$. To capture this formally, we introduce the notion of combined definers, which describes how definers are introduced, and the notion of a subsuming context of a definer, to denote when definers in $Res_{\mathcal{ALC}}^{norm\prec_l}(\mathcal{N} \cup \mathcal{M})$ have corresponding definers in $Res_{\mathcal{ALC}}^{norm\prec_l}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$.

Definition 4.4.4. Let $\mathcal{N}$ be any clause set. Two definers D_1 and D_2 are *combined in $\mathcal{N}$ by D*, if there are two clauses $\neg D \sqcup D_1, \neg D \sqcup D_2 \in \mathcal{N}$.

A definer D_1 *occurs in $\mathcal{N}$ in a subsuming context to a definer D_2* iff $D_1 \xrightarrow{c} D_2$, where $\xrightarrow{c}$ is the smallest relation such that $D_1 \xrightarrow{s} D_2$ if for every clause of the form $C \sqcup \mathsf{Q}r.D_2 \in \mathcal{N}$, $\mathsf{Q} \in \{\exists, \forall\}$, one of the following holds:

1. $C \sqcup \mathsf{Q}r.D_1 \in \mathcal{N}$, or

2. $C = \neg D_2' \sqcup C' \sqcup \mathsf{Q}' r.D_1$, $\mathsf{Q}' \in \{\exists, \forall\}$, and there is a definer D_2' with $D_1' \xrightarrow{c} D_2'$.

Let D_1 be a definer occurring in $\mathcal{N}_1$ and D_2 be a definer occurring in $\mathcal{N}_2$. D_1 occurs in a subsuming context to D_2 if D_1 occurs in $\mathcal{N}_1 \cup \mathcal{N}_2$ in a subsuming context to D_2.

Note that if a D_1 occurs in a subsuming context to D_2, and both are unsatisfiable, we obtain similar clauses when applying the $\exists$-elimination rule, where definers in negative definer literals might be replaced by definers in subsuming contexts. This will be used to show that every sequence of $\exists$-elimination rule applications that is used to infer the empty clause from $\mathcal{N} \cup \mathcal{M}$ has corresponding inferences in $\mathcal{N}^*$ and $Res^{norm \prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$.

In order to better structure the proofs on definers that will follow, we further introduce the notion of the root distance of a definer.

Definition 4.4.5. Let $\mathcal{N}$ be a set of clauses and D be a definer occurring in $\mathcal{N}$. The *root distance of* D, in symbols $rd(D)$, is the smallest number n such that there is a sequence of clauses $C \sqcup \mathsf{Q} r_1.D_1$, $\neg D_1 \sqcup C_1 \sqcup \mathsf{Q} r_2.D_2$, ..., $\neg D_{n-1} \sqcup C_{n-1} \sqcup \mathsf{Q} r_n.D \in \mathcal{N}$, where C does not contain a negative definer literal.

If $\mathcal{N}$ is the result of transforming an ontology $\mathcal{O}$ without definer symbols into normal form, the root distance of D is always defined, since each definer represents some concept occurring in $\mathcal{O}$, and there can be no infinite nesting of these concepts. The same holds if $\mathcal{N}$ is the saturation of such a clause set with any calculus introduced so far, since definers are introduced with clauses that give an upper bound on their root distance. Every introduced definer D_{12} is introduced via an application of a role propagation rule on two clauses of the form $C_1 \sqcup \forall r.D_1$ and $C_2 \sqcup \mathsf{Q} r.D_2$, $\mathsf{Q} \in \{\forall, \exists\}$, resulting in the clause $C_1 \sqcup C_2 \sqcup \mathsf{Q} r.D_{12}$. $rd(D_{12}) \leq rd(D) + 1$, where D is any definer occurring in a negative definer literal in C_1 or C_2.

The main challenge for proving interpolation completeness involves showing that introduced definers involved in showing unsatisfiability of $\mathcal{N} \cup \mathcal{M}$ have corresponding definers that are used in the computation of $\mathcal{N}^*$ and $Res^{norm \prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$. The following lemma establishes this for introduced definers that represent conjunctions over definers from $\mathcal{N}$.

Lemma 4.4.6. *Let D_1 and D_2 be two definers occurring in $Res^{norm}_{\mathcal{ALC}}(\mathcal{N})$ that are combined in $Res^{norm}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ by a definer D_3. Then, D_1 and D_2 are combined by a*

definer D_3' in $\mathcal{N}^*$, and either D_3' occurs in a subsuming context to D_3, or there is a clause $\neg D_3'' \sqcup D_3' \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$, and D_3'' occurs in a subsuming context to D_3.

Proof. Let D_1, D_2 and D_3 be as in the lemma. Note that two definers D_1 and D_2 only get combined by applications of a role propagation rule on role restrictions over D_1 and D_2. Observe that D_1 and D_2 must occur under role restrictions for the same role. If they do not, no sequence of role propagations can lead to a rule application involving both D_1 and D_2, and hence, they are not combined in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$.

If D_1 and D_2 are combined in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$ by D_3, then they are also combined in $\mathcal{N}^*$ by D_3, and the lemma trivially holds. Assume D_1 and D_2 are not combined in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$. There are two possibilities. (1) They both occur under a universal role restriction. In this case, there are not combined because $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$ does not have the $\forall\forall$-role propagation rule. (2) They occur under role restrictions in clauses with different negative definer literals. In this case, role propagation is not applicable due to the restriction of $Res^{norm_{\prec_l}}_{\mathcal{ALC}}$ that no clause with more than one negative definer literal is inferred. If neither (1) nor (2) are satisfied, we can apply the $\forall\exists$-role propagation rule on the role restrictions containing D_1 and D_2, and D_1 and D_2 are combined in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$.

We distinguish the cases based on whether (2) is satisfied or not.

1. D_1 and D_2 occur under role restrictions in clauses C_1, $C_2 \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$ such that C_1 and C_2 do not contain different negative definer literals. Since D_1 and D_2 are not combined in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$ by D_3, they must both occur under a universal role restriction, as observed earlier. The only way in which they can then be combined in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ is due to a clause $C_3 \sqcup \exists r.D_4 \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$. The combination of D_1 and D_2 then takes place due to the following inferences:

   1. $C_1 \sqcup \forall r.D_1$
   2. $C_2 \sqcup \forall r.D_2$
   3. $C_3 \sqcup \exists r.D_4$
   4. $C_1 \sqcup C_3 \sqcup \forall r.D_{14}$ *($\forall\exists$-role propagation 1, 3)*
   5. $\neg D_{14} \sqcup D_1$ *($D_{14} \sqsubseteq D_1$)*
   6. $\neg D_{14} \sqcup D_4$ *($D_{14} \sqsubseteq D_4$)*

7. $C_1 \sqcup C_2 \sqcup C_3 \sqcup \exists r.D_3$ *(∀∃-role propagation 4, 2)*
8. $\neg D_3 \sqcup D_{14}$ *($D_3 \sqsubseteq D_{14}$)*
9. $\neg D_3 \sqcup D_2$ *($D_3 \sqsubseteq D_{14}$)*
10. $\neg D_3 \sqcup D_1$ *(Resolution 5, 8)*

Using $Int_{\mathcal{ALC}}$, the ∀∀-role propagation rule can be applied on C_1 and C_2, and we obtain the following clauses that are in $\mathcal{N}^*$:

1. $C_1 \sqcup \forall r.D_1$
2. $C_2 \sqcup \forall r.D_2$
3. $C_1 \sqcup C_2 \sqcup \forall r.D_3'$ *(∀∃-role propagation 1, 2)*
4. $\neg D_3' \sqcup D_1$ *($D_3' \sqsubseteq D_1$)*
5. $\neg D_3' \sqcup D_2$ *($D_3' \sqsubseteq D_2$)*

$Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$ then contains the following clauses:

6. $C_3 \sqcup \exists r.D_4$
7. $C_1 \sqcup C_2 \sqcup C_3 \sqcup \exists r.D_3''$ *(∀∃-role propagation 3, 6)*
8. $\neg D_3'' \sqcup D_4$ *($D_3'' \sqsubseteq D_4$)*
9. $\neg D_3'' \sqcup D_3'$ *($D_3'' \sqsubseteq D_3'$)*
10. $\neg D_3'' \sqcup D_1$ *(Resolution 4, 9)*
11. $\neg D_3'' \sqcup D_2$ *(Resolution 5, 9)*

D_1 and D_2 are combined by D_3'' in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$, and D_3'' occurs in a subsuming context to D_3.

2. D_1 and D_2 occur under role restrictions in clauses in $C_1, C_2 \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$ such that C_1 and C_2 contain two different negative definers literals $\neg D_1'$ and $\neg D_2'$. Let $C_1 = \neg D_1' \sqcup C_1' \sqcup \mathsf{Q}r.D_1$ and $C_2 = \neg D_2' \sqcup C_2' \sqcup \mathsf{Q}r.D_1$. D_1 and D_2 are combined in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ by a definer D_3. Since definers only get combined via applications of a role propagation rule, there must be a clause $C_3 \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ which is of the form $\neg D_3' \sqcup C_3' \sqcup \mathsf{Q}r.D_3$, and $\neg D_3' \sqcup D_1'$, $\neg D_3' \sqcup D_2' \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$. That is, D_1' and D_2' are combined in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ by D_3'. Otherwise, since $Res^{norm}_{\mathcal{ALC}}$ infers only clauses with

at most one negative definer literal, D_1 and D_2 cannot get combined by any sequence of role propagations. For the same reason, D_1 and D_2 can only be combined in $Res^{norm}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$ if D'_1 and D'_2 are combined by a definer D''_3 in $Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$. If additionally D''_3 occurs in $Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$ in a subsuming context to D'_3, the lemma holds for D_1 and D_2, either because we can apply $\forall\exists$-role propagation, or a similar sequence of rule applications as in Case 1. In other words, whether the lemma holds for D_1 and D_2 is solely determined by whether it holds for D'_1 and D'_2. For D'_1 and D'_2, again either Case 1 or Case 2 applies. If we can reduce the situation for D_1 and D_2 after a sequence of finitely many steps to a situation for a pair of definers D^0_1 and D^0_2 on which Case 1 applies, we are done. Note that $rd(D'_1) \leq rd(D_1 - 1)$ and $rd(D'_2) \leq rd(D_2 - 1)$. This means, after at most $rd(D_1)$ steps, we arrive at a pair of definers D^0_1 and D^0_2 for which we have a situation as in Case 1. We obtain that D_1 and D_2 are combined by a definer D'_3 in $Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$ that occurs in a subsuming context to D_3.

We have shown that in both cases, D_1 and D_2 are combined by a definer D'_3 in $Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$, and D'_3 occurs in a subsuming context to D_3. □

If we look at the root distance of definers occurring in $\mathcal{M}$, or that are combined from at least one definer from $\mathcal{M}$, we find an important property which is due to how $\mathcal{M}$ is generated.

Lemma 4.4.7. *Let $\neg D_1 \sqcup C \sqcup \mathsf{Q}r.D_2$ be a clause occurring in $\mathcal{N}_i$, where $\mathcal{N}_i = Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ or $\mathcal{N}_i = Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$, such that either D_2 occurs in $\mathcal{M}$ or $\neg D_2 \sqcup D \in \mathcal{N}_i$, where D occurs in $\mathcal{M}$. Then, $rd(D_2) = rd(D_1) + 1$.*

Proof. Note that for clause sets $\mathcal{N}'$ that are the direct result of normalisation, $\neg D_1 \sqcup C \sqcup \mathsf{Q}r.D_2 \in \mathcal{N}'$ always implies $rd(D_2) = rd(D_1)+1$, due to how definers are introduced by the normalisation. Note that because positive definer literals only occur in clauses of the form $\neg D' \sqcup D$ (Lemma 4.2.5), saturation cannot decrease the root distance of a definer. Saturation also cannot increase the root distance of a definer. The lemma therefore holds for definers D_2 occurring in $\mathcal{M}$. Assume D_2 is a definer such that $\neg D_2 \sqcup D \in \mathcal{N}_i$, where $\mathcal{N}_i$ is as in the lemma and D occurs in $\mathcal{M}$.

We note the special character of the clause set $\mathcal{M}$: $\mathcal{M} = Cl(\exists r^*.(C \sqcap \neg D))$, and therefore every clause in $\mathcal{M}$ is either of the form $\exists r^*.D^*$ or of the form $\neg D \sqcup C$. Since r^* does not occur in $\mathcal{N}$, any non-empty clause that is inferred from a clause in $\mathcal{M}$ using role propagation will contain at least one negative definer as well. New definers only get introduced via the role propagation rule. Therefore, D_2 must have been introduced via an application of the role propagation rule, where at least one premise contains a negative definer literal. Assume that D_2 is such that it is introduced via role propagation on the clauses $\neg D_1 \sqcup C_1 \sqcup \mathsf{Q}r.D_2'$ and $C_2 \sqcup \mathsf{Q}r.D_2''$, where the lemma holds for D_1, D_2' and D_2''. $Res_{\mathcal{ALC}}^{norm \prec_l}$ only infers clauses with at most one negative definer literal, and therefore C_2 does not contain any negative definer literal other than $\neg D_1$. Furthermore, as observed earlier, no inference can decrease the root distance of a definer. We obtain $rd(D_2) = rd(D_2') = rd(D_1{+}1)$. From this it follows that no sequence of rule applications can lead to the introduction of a definer D_2 that is as in the lemma and that occurs in a clause $\neg D_1 \sqcup C \sqcup \mathsf{Q}r.D_2$ such that $rd(D_2) \neq rd(D_1) + 1$. □

Now that we established some properties about introduced definers, we can focus on inferences on clauses that contain an introduced definer.

We can show that inferences on symbols outside $\mathcal{S}$ which are applied when computing $Res_{\mathcal{ALC}}^{norm}(\mathcal{N} \cup \mathcal{M})$ have corresponding inferences that are applied when computing $\mathcal{N}^*$ and $Res_{\mathcal{ALC}}^{norm}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$.

Lemma 4.4.8. *Let $C \in Res_{\mathcal{ALC}}^{norm \prec_l}(\mathcal{N} \cup \mathcal{M})$ be the result of an inference on a literal not in $\mathcal{S} \cup N_d$. Then, either $C \in \mathcal{N}^*$, or $\mathcal{N}^*$ contains a clause C' that is the result of replacing a negative definer literal $\neg D$ in C by a negative definer literal $\neg D'$, and D' occurs in a subsuming context to D or $\neg D'' \sqcup D' \in Res_{\mathcal{ALC}}^{norm \prec_l}(\mathcal{N}^* \cup \mathcal{M})$, where D'' occurs in a subsuming context to D.*

Proof. Assume C is the result of an inference on two clauses C_1 and C_2, where for the maximal literal L_1 in C_1 we have $sig(L_1) \not\subseteq \mathcal{S} \cup N_d$. If $C \in Res_{\mathcal{ALC}}^{norm \prec_l}(\mathcal{N})$, $C \in \mathcal{N}^*$ follows trivially. Assume therefore $C \notin Res_{\mathcal{ALC}}^{norm \prec_l}(\mathcal{N})$. It follows $C_1 \notin Res_{\mathcal{ALC}}^{norm \prec_l}(\mathcal{N})$ or $C_2 \notin Res_{\mathcal{ALC}}^{norm \prec_l}(\mathcal{N})$. $Res_{\mathcal{ALC}}^{norm \prec_l}$ prefers inferences on symbols that are not in $\mathcal{S}$, or with clauses of the form $\neg D_1 \sqcup D_2$. If C contains no negative definer literal, it can therefore be inferred solely starting from the clauses in $\mathcal{N}$, which contradicts $C \notin Res_{\mathcal{ALC}}^{norm \prec_l}(\mathcal{N})$. The same holds if C contains a negative definer literal $\neg D$

where D occurs in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$. C is therefore of the form $\neg D \sqcup C'$, where D occurs in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ but not in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$.

Since $\mathcal{M}$ only contains symbols that are in $\mathcal{S} \cup N_d$, and at least one of C_1 and C_2 contains a symbol that is not in $\mathcal{S} \cup N_d$, C_1 or C_2 must be the result of inferences involving clauses in $\mathcal{N}$ with symbols outside $\mathcal{S} \cup N_d$. These inferences only involve clauses of the form $\neg D' \sqcup D''$ and symbols that are not in $\mathcal{S}$. Therefore, there must be two definers D_1 and D_2 in $\mathcal{N}$ such that $\neg D \sqcup D_1$, $\neg D \sqcup D_2 \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$. By Lemma 4.4.6, there are then two clauses $\neg D' \sqcup D_1$, $\neg D' \sqcup D_2 \in \mathcal{N}^*$, and either D' occurs in a subsuming context to D, or $\neg D'' \sqcup D' \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$, where D'' occurs in a subsuming context to D. From this lemma also follows that for every definer D_i occurring in $\mathcal{M}$ for which $\neg D \sqcup D_i \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$, $\neg D' \sqcup D_i \in Int_{\mathcal{ALC}}(\mathcal{N})$. Note that this definer D' satisfies the conditions given in the lemma.

Due to the ordering, C must be inferred in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ starting with inferences on these clauses $\neg D \sqcup D_i$ and clauses from $\mathcal{N}$, followed by inferences on symbols not in $\mathcal{S}$ that have at most one negative definer literal. Since for every $\neg D \sqcup D_i \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ we have $\neg D' \sqcup D_i \in \mathcal{N}^*$, we obtain that $\neg D' \sqcup C'$ can be inferred in $\mathcal{N}^*$. $\neg D' \sqcup C'$ is the result of replacing $\neg D$ in C by a negative definer literal which is as specified in the lemma. □

Lemma 4.4.9. *For every clause $\neg D \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$, there is a clause $\neg D'$ in $\mathcal{N}^*$ or in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$, and D' occurs in a subsuming context to D.*

Proof. For definers D with $\neg D \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$ or $\neg D \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{M})$, the lemma follows easily. Assume therefore $\neg D \notin Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$ and $\neg D \notin Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{M})$. As we observed in the proof for Lemma 4.4.8, clauses in $\mathcal{N}$ can contribute to the unsatisfiability of definers occurring in $\mathcal{M}$, but clauses in $\mathcal{M}$ cannot contribute to the unsatisfiability of definers occurring in $\mathcal{N}$. Therefore, D is either a definer from $\mathcal{M}$, or there is a clause $\neg D \sqcup D' \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$, where D' occurs in $\mathcal{M}$.

The proof is by contradiction, where we make use of the structure given by the root distance defined in Definition 4.4.5. Let $\prec^r_d$ be a total ordering on the definer symbols in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ such that $D_1 \prec^r_d D_2$ if $rd(D_1) < rd(D_2)$. Assume D is the largest unsatisfiable definer in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ according to $\prec^r_d$ such that D does not occur in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N})$, and for which there is no unsatisfiable definer D' in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N}^*_{\mathcal{S}} \cup \mathcal{M})$

in a subsuming context to D. By Lemma 4.4.6, $Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^*_{\mathcal{S}} \cup \mathcal{M})$ contains a definer D' that occurs in a subsuming context to D. By Lemma 4.4.8, there is furthermore such a definer such that for every clause $\neg D \sqcup C \in Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ that is the result of an inference on a literal not in $\mathcal{S}$, there is a corresponding clause $\neg D' \sqcup C \in Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$. Let D' be such a definer.

Let $\prec_c$ be a total order on clauses based on the multiset extension of $\prec_l$, such as we used it in the proof for refutational completeness of $Res_{\mathcal{ALC}}$ in Section 4.2.3. Let $\neg D \sqcup C \in Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ be the *largest* clause according to $\prec_c$ such that $\neg D' \sqcup C \notin Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$. By Lemma 4.4.8, $\neg D \sqcup C$ is not the result of an inference on a literal not in $\mathcal{S} \cup N_d$. Therefore, $\neg D \sqcup C$ is the result of an inference on a literal in $\mathcal{S} \cup N_d$. We distinguish the cases based on how $\neg D \sqcup C$ is derived.

1. $\neg D \sqcup C$ is inferred via resolution or $\forall\exists$-role propagation on two clauses $\neg D \sqcup C_1$ and $\neg D \sqcup C_2$. Since inferences are only performed on maximal literals, $\neg D \sqcup C$ is smaller than both clauses according to $\prec_c$. Because by assumption $\neg D' \sqcup C \notin Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$, at least one of $\neg D' \sqcup C_1$ or $\neg D' \sqcup C_2$ does not occur in $Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$ either. But this contradicts the assumption that $\neg D \sqcup C$ is the largest such clause.

2. $\neg D \sqcup C$ is inferred via $\exists$-elimination on two clauses $\neg D \sqcup C \sqcup \exists r.D_1$ and $\neg D_1$. Because $\neg D' \sqcup C$ does not occur in $Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$, neither does $\neg D' \sqcup C \sqcup \exists r.D_1$ nor $\neg D_1$. Therefore, D_1 has not been introduced solely by clauses in $\mathcal{N}$, and there must be a clause $\neg D_1 \sqcup D \in Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ such that D occurs in $\mathcal{M}$. By Lemma 4.4.7, we then have $rd(D_1) = rd(D) + 1$ and $D \prec^r_d D_1$. By assumption, D is the largest definer according to $\prec^r_d$ that does not fulfil the lemma. Therefore, there is a definer D'_1 with $\neg D'_1 \in Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$ and D'_1 occurs in a subsuming context to D_1. We obtain $\neg D' \sqcup C \sqcup \exists r.D'_1 \in Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$, and hence, also $\neg D' \sqcup C \in Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$. This contradicts our assumption that $\neg D' \sqcup C \notin Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$

We obtain that there is no largest clause according to $\prec_c$ such that $\neg D' \sqcup C \notin Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$, and hence, $\neg D' \in Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$. This contradicts the assumption that there is a largest unsatisfiable definer D according to $\prec^r_d$ such that D does not occur in $Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N})$ and for which there is no unsatisfiable definer D' in

$Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$ in a subsuming context to D. Therefore, for every definer D with $\neg D \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$, there is such a definer D'. □

We can finally establish the lemma for which we introduced the clause set $\mathcal{M}$.

Lemma 4.4.10. *$\mathcal{N} \cup \mathcal{M}$ is unsatisfiable iff $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ is unsatisfiable.*

Proof. The lemma holds obviously for the case where $\mathcal{M}$ is unsatisfiable. Assume $\mathcal{M}$ is unsatisfiable. We first prove that $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ is unsatisfiable if $\mathcal{N} \cup \mathcal{M}$ is unsatisfiable. We distinguish two cases.

1. $\mathcal{N}$ is unsatisfiable. Due to the constraints on the ordering $\prec_l$, for every clause C that is not completely in $\mathcal{S} \cup N_d$, inferences in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}$ only apply on literals in C that are also not in $\mathcal{S} \cup N_d$. After all these inferences have been performed, we can discard these clauses. Observe that all remaining inferences are also possible from $\mathcal{N}^{\mathcal{S}}$. Therefore, if $\mathcal{N}$ is unsatisfiable, $\bot \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}})$.

2. $\mathcal{N}$ is satisfiable. Then, the clauses in $\mathcal{M}$ must contribute to inferring an inconsistency of $\mathcal{N} \cup \mathcal{M}$. Every clause in $\mathcal{M}$ is either of the form $\exists r^*.D^*$ or of the form $\neg D \sqcup C$, where r^*, D^* and D do not occur in $\mathcal{N}$. Therefore, the only way in which the empty clause can be inferred from $\mathcal{N} \cup \mathcal{M}$ if $\mathcal{N}$ is satisfiable, is by applying the $\exists$-elimination rule on two clauses $\exists r^*.D^*$ and $\neg D^*$ in $Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$. By Lemma 4.4.9, if $\neg D^* \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$, then also $\neg D^* \in Res^{norm_{\prec_l}}_{\mathcal{ALC}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$. We obtain that if $\mathcal{N} \cup \mathcal{M}$ is unsatisfiable, so is $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$.

For the other direction, assume $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ is unsatisfiable. Since $Int_{\mathcal{ALC}}$ is sound, $\mathcal{N}^{\mathcal{S}}$ only contains sound inferences from $\mathcal{N}$. Therefore, if $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ is unsatisfiable, so is $\mathcal{N} \cup \mathcal{M}$. We obtain that $\mathcal{N} \cup \mathcal{M}$ is unsatisfiable iff $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ is unsatisfiable. □

Because our construction started from a clause set $\mathcal{M}$ representing $\exists r^*(C \sqcap \neg D)$, where C and D are arbitrary $\mathcal{ALC}$ concepts with $sig(C \sqsubseteq D) \subseteq \mathcal{S}$, we obtain that $\mathcal{O} \models C \sqsubseteq D$ iff $\mathcal{O}^{\mathcal{S}} \models C \sqsubseteq D$, where $sig(C \sqsubseteq D) \subseteq \mathcal{S}$. This allows us to establish interpolation completeness of $Int_{\mathcal{ALC}}$.

Theorem 4.4.11. *$Int_{\mathcal{ALC}}$ is interpolation complete for $\mathcal{ALC}\nu$.*

We note that in order to prove this result we exploited three major properties of the calculi $Res_{\mathcal{ALC}}$ and $Int_{\mathcal{ALC}}$. (1) We preserve refutational completeness if we restrict

$Res_{\mathcal{ALC}}$ to inferences of clauses with at most one negative definer literal. (2) We preserve refutational completeness if we additional restrict $Res_{\mathcal{ALC}}$ with an ordering that makes literals outside of a given signature larger than other literals. (3) The more complex property stated in Lemma 4.4.6 that any pair of definers from $\mathcal{N}$ that are combined by a definer D in $Res^{norm\prec_l}_{\mathcal{ALC}})(\mathcal{N} \cup \mathcal{M})$ are combined by a definer D' in $\mathcal{N}^* = Int_{\mathcal{ALC}}(\mathcal{N})$, and D' contributes to a definer in $Res^{norm\prec_l}_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$ that occurs in a subsuming context to D. This last lemma motivates the last modification of the calculus we introduced in this chapter, the extension of $Res^{norm}_{\mathcal{ALC}}$ by the $\forall\forall$-role propagation rule, and methods based on similar calculi to $Res_{\mathcal{ALC}}$ will need a corresponding property.

4.4.3 Minimising Computed Uniform Interpolants

From the perspective of practicality, computing the saturated set $Int_{\mathcal{ALC}}(\mathcal{N})$ completely is not feasible, and not necessary, to compute a uniform interpolant, since it contains more entailments than are needed. $Int_{\mathcal{ALC}}(\mathcal{N})$ can be seen more as an upper bound of the clausal representation of the uniform interpolant. While a more detailed description on how to compute uniform interpolants in practice is given in Chapter 8, we can already give a descriptive definition of a more tight set of clauses that is sufficient to represent a uniform interpolant in clausal form.

Definition 4.4.12. Let $\mathcal{N}$ be a set of clauses and $\mathcal{S}$ a signature. The *minimal clausal representation of the $\mathcal{ALC}\nu$ uniform interpolant of $\mathcal{N}$ for $\mathcal{S}$*, which we denote by $Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min}$, is the smallest subset of $Int_{\mathcal{ALC}}(\mathcal{N})$ such that every clause $C \in Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min}$ satisfies $sig(C) \subseteq \mathcal{S} \cup N_d$ and is not redundant with respect to $Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min}$, and that contains all clauses from $Int_{\mathcal{ALC}}(\mathcal{N})$ that additionally satisfy one of the following conditions.

1. $C \in \mathcal{N}$.
2. C is the result of applying a rule of $Int_{\mathcal{ALC}}$ on two clauses $C_1, C_2 \in Int_{\mathcal{ALC}}(\mathcal{ALC})$ with $sig(C_1) \cap (\mathcal{S} \cup N_d) \neq \emptyset$ or $sig(C_2) \cap (\mathcal{S} \cup N_d) \neq \emptyset$.
3. C contains a definer D which occurs in another clause of the form $\neg D \sqcup C'$ in $Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min}$.

Condition 1 ensures that all clauses from the input set $\mathcal{N}$ are kept which are in the desired signature. Condition 2 ensures that all entailments that are in the desired signature, and are directly derived from clauses in $Int_{\mathcal{ALC}}(\mathcal{N})$ that are not in the desired signature, are kept. Note that, if we are only forgetting concept symbols, these are all conclusions of the resolution rule, and if we are forgetting only role symbols, these are all conclusions of the $\exists$-elimination rule.

Condition 3 ensures that $Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min}$ is closed under definer-occurrences. For example, if $Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min}$ contains a clause $C_1 = \neg D \sqcup C_1'$, then it should also contain all clauses of the form $C_2 = C_2' \sqcup \mathsf{Q}r.D$, $\mathsf{Q} \in \{\forall, \exists\}$. Otherwise, C_1 would not contribute to the result of undoing the structural transformation.

If $\mathcal{N} = Cl(\mathcal{O})$ for an $\mathcal{ALC}$ ontology $\mathcal{O}$, then $Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min}$ is sufficient to compute an $\mathcal{ALC}\nu$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$, as the following theorem shows.

Theorem 4.4.13. *Let $\mathcal{O}$ be an $\mathcal{ALC}\nu$ ontology, $\mathcal{S}$ be a signature and $\mathcal{N} = Cl(\mathcal{O})$. Then, $\mathcal{O}^{\mathcal{S}} = Ont(Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min})$ is an $\mathcal{ALC}\nu$ uniform interpolant of $\mathcal{O}$.*

Proof. Since $Int_{\mathcal{ALC}}$ is interpolation complete, $\mathcal{O}^{\mathcal{S}} = Ont(Int_{\mathcal{ALC}}(\mathcal{N})_{\mathcal{S}})$ is an $\mathcal{ALC}\nu$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$. $Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min}$ contains all clauses in $Int_{\mathcal{ALC}}(\mathcal{N})$ that are inferences on symbols outside of $\mathcal{S}$ (Condition 2), and all clauses in the desired signature that are already present in $\mathcal{N}$ (Condition 1). Condition 3 ensures additionally that $Cl(O^{\mathcal{S}})$ contains the same definers as $Int_{\mathcal{ALC}}(\mathcal{N}, \mathcal{S})_{min}$ modulo renaming. Hence, it is possible to compute $Int_{\mathcal{ALC}}(\mathcal{N})_{\mathcal{S}}$ modulo definer names by saturating $Cl(O^{\mathcal{S}})$ using $Int_{\mathcal{ALC}}$. We obtain that all entailments of $Ont(Int_{\mathcal{ALC}}(\mathcal{N})_{\mathcal{S}})$ are preserved, and since $\mathcal{O}^{\mathcal{S}}$ is also in the desired signature, it is an $\mathcal{ALC}$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$. □

We conclude the chapter with some examples illustrating the uniform interpolation procedure.

4.5 Examples

Example 4.5.1 (Forgetting a concept symbol)**.** $\mathcal{O}_5$ is the following ontology:

$$A \sqsubseteq B \sqcup C$$

$$B \sqsubseteq \exists r.B$$

$$C \sqsubseteq \forall r.\neg B$$

We want to compute the uniform interpolant for $\mathcal{S}_5 = \{A, C, r\}$. Normalisation results in the following clause set $\mathcal{N}_5 = Cl(\mathcal{O}_5)$.

1. $\neg A \sqcup B \sqcup C$ *(Normal Form Transformation)*
2. $\neg B \sqcup \exists r.D_1$ *(Normal Form Transformation)*
3. $\neg D_1 \sqcup B$ *(Normal Form Transformation)*
4. $\neg C \sqcup \forall r.D_2$ *(Normal Form Transformation)*
5. $\neg D_2 \sqcup \neg B$ *(Normal Form Transformation)*

According to Condition 2 in Definition 4.4.12, we need to compute all resolvents on B that can be derived from $\mathcal{N}_5$ using $Int_{\mathcal{ALC}}$, modulo redundant clauses. Note that this excludes clauses with more than one negative definer literal, since these are not inferred by $Int_{\mathcal{ALC}}$. We first apply resolution.

6. $\neg A \sqcup C \sqcup \exists r.D_1$ *(Resolution 1, 2)*

7. $\neg D_2 \sqcup \neg A \sqcup C$ *(Resolution 1, 5)*

8. $\neg D_1 \sqcup \exists r.D_1$ *(Resolution 2, 3)*

We cannot resolve on Clauses 3 and 5, since the conclusion has more than one negative definer literal. Theorem 4.3.3 shows that these inferences are not necessary if we only want to derive normal clauses. We can however apply $\forall\exists$-role propagation on Clause 2 and 4, which makes further applications of the resolution rule possible.

~~9.~~ $\neg B \sqcup \neg C \sqcup \exists r.D_{12}$ *($\forall\exists$-Role Propagation 2, 4)*

~~10.~~ $\neg D_{12} \sqcup D_1$ *($D_{12} \sqsubseteq D_1$)*

~~11.~~ $\neg D_{12} \sqcup D_2$ *($D_{12} \sqsubseteq D_2$)*

~~12.~~ $\neg D_{12} \sqcup B$ *(Resolution 3, 10)*

~~13.~~ $\neg D_{12} \sqcup \neg B$ *(Resolution 5, 11)*

14. $\neg D_{12}$ *(Resolution 12, 13)*

Clause 14 subsumes Clauses 10–13, and makes $\exists$-elimination on Clause 9 possible.

15. $\neg B \sqcup \neg C$ *($\exists$-Elimination 9, 14)*

Clause 15 subsumes Clause 9. We saturate the remaining clauses.

~~16.~~ $\neg A \sqcup C \sqcup \neg C$ *(Resolution 1, 15, Tautology)*

17. $\neg D_1 \sqcup \neg C$ *(Resolution 3, 15)*

In $Int_{\mathcal{ALC}}(\mathcal{N}_5, \mathcal{S}_1)_{min}$, only clauses that neither contain B nor a positive definer are included. These are the Clauses 4, 6, 7, 8, 14 and 17 with a bold face number. After eliminating all definers, we obtain the following set of axioms.

$$\top \sqsubseteq \neg C \sqcup \forall r.(\neg A \sqcup C)$$
$$\top \sqsubseteq \neg A \sqcup C \sqcup \exists r.\nu X.(\exists r.X \sqcap \neg C)$$

This set of axioms can be represented equivalently by the following ontology $\mathcal{O}_5^{\mathcal{S}_5}$.

$$A \sqsubseteq C \sqcup \exists r.\nu X.(\exists r.X \sqcap \neg C)$$
$$C \sqsubseteq \forall r.(\neg A \sqcup C)$$

$\mathcal{O}_5^{\mathcal{S}_5}$ is an $\mathcal{ALC}$ uniform interpolant of $\mathcal{O}_5$ for $\mathcal{S}_5$.

Example 4.5.2 (Forgetting a role symbol)**.** $\mathcal{O}_6$ contains the following axioms.

$$A \sqsubseteq \exists r.(A \sqcup B)$$
$$B \sqsubseteq \forall r.\neg A$$
$$C \sqsubseteq \forall r.\neg B$$

We want to compute a uniform interpolant for $\mathcal{S}_6 = \{A, B, C\}$, that is, we want to forget r.

Normalisation results in the following clausal representation $\mathcal{N}_6 = Cl(\mathcal{O}_6)$:

1. $\neg A \sqcup \exists r.D_1$ *(Normal Form Transformation)*

2. $\neg D_1 \sqcup A \sqcup B$ *(Normal Form Transformation)*

3. $\neg B \sqcup \forall r.D_2$ *(Normal Form Transformation)*

4. $\neg D_2 \sqcup \neg A$ *(Normal Form Transformation)*

5. $\neg C \sqcup \forall r.D_3$ *(Normal Form Transformation)*

6. $\neg D_3 \sqcup \neg B$ *(Normal Form Transformation)*

According to Condition 2 in Definition 4.4.12, we have to infer all conclusions of the $\exists$-elimination rule on r. In order to do so, we have to infer all unary clauses of the form $\neg D$, $D \in N_d$. This clause cannot be derived for any of the existing definers. Therefore, we have to trigger the introduction of new definers using the role propagation rules.

7. $\neg A \sqcup \neg B \sqcup \exists r.D_{12}$ *($\forall\exists$-Role Propagation 1, 3)*

8. $\neg D_{12} \sqcup D_1$ *($D_{12} \sqsubseteq D_1$)*

9. $\neg D_{12} \sqcup D_2$ *($D_{12} \sqsubseteq D_2$)*

~~10.~~ $\neg D_{12} \sqcup A \sqcup B$ *(Resolution 2, 8)*

11. $\neg D_{12} \sqcup \neg A$ *(Resolution 4, 9)*

12. $\neg D_{12} \sqcup B$ *(Resolution 10, 11)*

Clause 12 subsumes Clause 10. We cannot derive $\neg D_{12}$, but we can trigger the introduction of another definer D_{123} that is unsatisfiable.

13. $\neg A \sqcup \neg B \sqcup \neg C \sqcup \exists r.D_{123}$ *($\forall\exists$-Role Propagation 5, 7)*

~~14.~~ $\neg D_{123} \sqcup D_{12}$ *($D_{123} \sqsubseteq D_{12}$)*

~~15.~~ $\neg D_{123} \sqcup D_3$ *($D_{123} \sqsubseteq D_3$)*

~~16.~~ $\neg D_{123} \sqcup B$ *(Resolution 12, 14)*

~~17.~~ $\neg D_{123} \sqcup \neg B$ *(Resolution 6, 15)*

18. $\neg D_{123}$ *(Resolution 16, 17)*

Clause 18 subsumes all other clauses that contain the negative definer literal $\neg D_{123}$. In addition, it makes the application of the $\exists$-elimination rule possible.

19. $\neg A \sqcup \neg B \sqcup \neg C$ *($\exists$-Elimination 13, 18)*

Clause 19 is a clause in the desired signature $\mathcal{S}_6$, that has been inferred in one step from a clause in $Int_{\mathcal{ALC}}(\mathcal{N}_6)$ that is not in $\mathcal{S}_6$. Therefore, Clause 19 is part of the result set $Int_{\mathcal{ALC}}(\mathcal{N}_6, \mathcal{S}_6)_{min}$.

By applying the $\forall\forall$-role propagation rule on Clause 3 and 5, the introduction of a definer D_{23} representing $D_2 \sqcap D_3$ is triggered. However, this definer does not occur under an existential role restriction, and is therefore not immediately helpful. Applying the $\forall\exists$-rule on that clause and Clause 1 leads to a clause that has an existential role restriction for a definer D_{231} representing $D_{23} \sqcap D_1$. The definer D_{231} is not introduced as a new definer, if we follow the technique described in Section 4.2.2. Observe that $\mathit{conj}(D_{213}) = \{D_1, D_2, D_3\}$. We also have $\mathit{conj}(D_{123}) = \{D_1, D_2, D_3\}$, which means we would reuse this definer. Consequently, the conclusion of this derivation is Clause 13, which we already inferred. In fact, we already inferred all clauses in $\mathit{Int}_{\mathcal{ALC}}(\mathcal{N}_6, \mathcal{S}_6)_{min}$. These are the Clauses 2, 4, 6 and 18 that have a bold face number.

The only clause without a negative definer is Clause 18, and this clause does not contain any role restrictions. Elimination of the definers would therefore simply delete Clause 2, 4 and 6. Clause 18 is the sole axiom of the uniform interpolant:

$$\top \sqsubseteq \neg A \sqcup \neg B \sqcup \neg C$$

Alternatively, the uniform interpolant can be represented with the following equivalent axiom:

$$A \sqcap B \sqcap C \sqsubseteq \bot$$

This is a uniform interpolant $\mathcal{O}_6^{\mathcal{S}_6}$ of $\mathcal{O}_6$ for $\mathcal{S}_6$.

Chapter 5

Extending the Method to $\mathcal{SH}$ and $\mathcal{SIF}$

In the last chapter, we introduced the $\mathcal{ALC}$ normal form and the calculi $Res_{\mathcal{ALC}}$ and $Int_{\mathcal{ALC}}$, together with extensions for ordered resolution and redundancy elimination. In this chapter, we extend the normal form and the calculi step by step to incorporate role hierarchies (Section 5.1), transitive roles (Section 5.3) and functional role restrictions (Section 5.4). Finally, in Section 5.5, we present a calculus for the description logic $\mathcal{SIF}$ that uses all these constructs except for role hierarchies. In order to reason with the new expressivity, we introduce additional rules for each description logic. For inverse roles, we furthermore give restrictions on the order in which rules are applied. Functional role restrictions are simple forms of number restrictions, which are discussed in Chapter 6. Section 5.4 therefore already prepares some key ideas presented in Chapter 6.

After presenting the uniform interpolation method for $\mathcal{ALCH}$ in Section 5.1, we discuss its extensions $\mathcal{SH}$, $\mathcal{ALCHF}$ and $\mathcal{ALCHI}$ in isolation, since functional role restrictions interact with transitive and inverse roles in non-trivial ways. Furthermore, if a description logic supports both inverse roles and functional role restrictions, it loses the finite model property. This means, there are $\mathcal{SIF}$ ontologies that have only infinite models. For this reason, we have to adapt the model construction presented in Chapter 4, which always creates a finite interpretation. This adaptation is presented in Section 5.5, where refutational and interpolation completeness of the calculi for $\mathcal{SIF}$ is proved. We finish this chapter with an overview of all rules introduced in this chapter.

An overview of all calculi presented in the thesis can be found in the conclusion in Chapter 9.

In the literature, several techniques have been proposed to encode some of the additional constructs in $\mathcal{SIF}$ and $\mathcal{SH}$ into $\mathcal{ALC}$ ontologies using rewrite rules. For transitivity axioms, such a rewriting can be found in Tobies (2001); Schmidt and Hustadt (2003a), for inverse roles and functional roles, methods are presented in Calvanese et al. (1998); Carral et al. (2014). However, these rewriting rules introduce additional symbols and are developed with specific reasoning goals in mind, such as classification or satisfiability checking. It is therefore not straightforward whether and how these rewrite rules can be used to extend a uniform interpolation method for $\mathcal{ALC}$ to a method for $\mathcal{SIF}$. In contrast, our methods directly integrate the new expressivity into the calculi. This is also required since uniform interpolants have to preserve entailments using these constructs.

Whereas for $\mathcal{ALC}$ ontologies, all uniform interpolants can be represented finitely by extending the input language with fixpoint operators, it turns out that for some description logics considered in this chapter, this is not sufficient. In particular, if role hierarchies and transitive roles are involved, we can only compute uniform interpolants for signatures that contain all transitive roles of the input ontology, and if role hierarchies and functional role restrictions are involved, we can only compute uniform interpolants for signatures that include all role symbols of the role hierarchy. Whereas it is likely that further language extensions can solve this problem, we focus on the cases where uniform interpolants can be expressed using fixpoint operators as only extension. For this reason, we define two restricted versions of interpolation completeness, which we establish for the respective calculi for description logics with transitive roles, functional role restrictions, and both.

Definition 5.0.3. A calculus Calc is *interpolation complete for forgetting concepts* in a language $\mathcal{L}$, if for any $\mathcal{L}$ ontology $\mathcal{O}$ and any signature $\mathcal{S}$ with $N_r \subseteq \mathcal{S}$, $\mathit{Ont}((\mathcal{N}_n^*)_\mathcal{S})$ is an $\mathcal{L}$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$, where $\mathcal{N}^* = \mathsf{Calc}(\mathit{Cl}(\mathcal{O}))$.

A calculus Calc is *interpolation complete for forgetting intransitive roles and concepts* in a language $\mathcal{L}$, if for any $\mathcal{L}$ ontology $\mathcal{O}$ and any signature $\mathcal{S}$ such that $r \in \mathcal{S}$ for every role r with $\mathsf{trans}(r) \in \mathcal{O}$, $\mathit{Ont}((\mathcal{N}_n^*)_\mathcal{S})$ is an $\mathcal{L}$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$, where $\mathcal{N}^* = \mathsf{Calc}(\mathit{Cl}(\mathcal{O}))$.

∃-Monotonicity

$$\frac{C \sqcup \exists R.D \qquad R \sqsubseteq S}{C \sqcup \exists S.D}$$

∀-Monotonicity

$$\frac{C \sqcup \forall R.D \qquad S \sqsubseteq R}{C \sqcup \forall S.D}$$

Role Hierarchy

$$\frac{S \sqsubseteq R \qquad R \sqsubseteq T}{S \sqsubseteq T}$$

Figure 5.1: Additional inference rules in $Res_{\mathcal{ALCH}}$ and $Int_{\mathcal{ALCH}}$.

5.1 Role Hierarchies

5.1.1 $\mathcal{ALCH}$ without Redundancy Elimination

The description logic $\mathcal{ALCH}$ extends $\mathcal{ALC}$ with role inclusion and role equivalence axioms of the forms $r \sqsubseteq s$ and $r \equiv s$, where $r, s \in N_r$. Whereas $\mathcal{ALC}$ ontologies only have a TBox, in $\mathcal{ALCH}$, ontologies may additionally have a non-empty RBox that contains these types of axioms. RBox axioms are represented in the normal form by clauses of the form $r \sqsubseteq s$. This is sufficient, since role equivalence axioms $r \equiv s$ are logically equivalent to corresponding pairs of role inclusion axioms $r \sqsubseteq s$ and $s \sqsubseteq r$.

Definition 5.1.1. An $\mathcal{ALCH}$ *clause* is either an $\mathcal{ALC}$ clause or a role inclusion axiom. An ontology $\mathcal{N}$ is in $\mathcal{ALCH}$ *normal form* if every axiom in $\mathcal{N}$ is an $\mathcal{ALCH}$ clause.

$Res_{\mathcal{ALCH}}$ and $Int_{\mathcal{ALCH}}$ respectively extend $Res_{\mathcal{ALC}}$ and $Int_{\mathcal{ALC}}$ with the rules shown in Figure 5.1. $Res_{\mathcal{ALCH}}$ extends $Res_{\mathcal{ALC}}$ by the ∃-monotonicity rule, which is sufficient for refutational completeness. $Int_{\mathcal{ALCH}}$ extends $Int_{\mathcal{ALC}}$ by the ∃- and ∀-monotonicity, as well as by the role hierarchy rule.

Note that the rules are defined in a general way, in the sense that they apply also to roles of the form r^-, even though these are not part of the normal form. We use this representation throughout this chapter, in order to be able to reuse the rules as formulated for description logics that allow for inverse roles.

The monotonicity rules derive information based on $\mathcal{ALC}$ clauses and role inclusions. Whereas the $\exists$-monotonicity rule replaces roles by their super-roles, the $\forall$-monotonicity rule replaces roles by their sub-roles. The role hierarchy rule makes the transitive closure of the role inclusions explicit in the saturated clause set.

By iterative application starting from a clause $C \sqcup \exists r.D$, the $\exists$-monotonicity rule adds a clause $C \sqcup \exists s.D$ for every super-role s of r in the transitive closure of the role hierarchy. This is why the other two rules are not needed for refutational completeness, and are not included in $Res_{\mathcal{ALCH}}$. On the other hand, in case a role symbol is to be eliminated, all three rules are needed for interpolation completeness.

To prove refutational completeness of $Res_{\mathcal{ALCH}}$, the model construction presented in the last chapter can be used without further adaptations. Recall that $\sqsubseteq_{\mathcal{N}}$ denotes the reflexive-transitive closure of the partial ordering $\sqsubseteq$ induced by clauses of the form $r \sqsubseteq s$ in $\mathcal{N}$ (see Section 3.1).

Lemma 5.1.2. *Let $\mathcal{N}$ be any satisfiable set of $\mathcal{ALCH}$ clauses, and let $\mathcal{I}$ be an interpretation built using the same procedure as presented in Section 4.2.3, but based on the clause set $\mathcal{N}^* = Res_{\mathcal{ALCH}}(\mathcal{N}) \setminus \{r \sqsubseteq s \in Res_{\mathcal{ALCH}}(\mathcal{N})\}$. Then, $\mathcal{I} \models r \sqsubseteq s$ for every role inclusion $r \sqsubseteq_{\mathcal{N}} s$ in the transitive closure of the role hierarchy.*

Proof. Due to the $\exists$-monotonicity rule, for every clause of the form $C \sqcup \exists r.D \in \mathcal{N}^*$ and every role s with $r \sqsubseteq_{\mathcal{N}} s$, there is a corresponding clause $C \sqcup \exists s.D \in \mathcal{N}^*$. Since these clauses determine which literals are added to the model fragments, for every literal $\exists r.D' \in I^D$ and every $r \sqsubseteq_{\mathcal{N}} s$, there is also the literal $\exists s.D' \in I^D$. Literals of this form determine which edges are added to the model $\mathcal{I}$. Therefore, if there is an edge $(x_{D_1}, x_{D_2}) \in r^{\mathcal{I}}$ and $r \sqsubseteq_{\mathcal{N}} s$, then also $(x_{D_1}, x_{D_2}) \in s^{\mathcal{I}}$. This establishes that $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$ whenever $r \sqsubseteq_{\mathcal{N}} s$, and therefore that $\mathcal{I} \models r \sqsubseteq s$ for every $r \sqsubseteq_{\mathcal{N}} s$. □

Theorem 5.1.3. *$Res_{\mathcal{ALCH}}$ is sound, terminating and refutationally complete.*

Proof. Termination follows from termination of $Res_{\mathcal{ALC}}$ (Lemma 4.2.4), since the introduction of definer symbols has not changed. Soundness of the $\exists$-monotonicity rule can be easily verified as well. For refutational completeness, assume that $\mathcal{N}$ is any satisfiable set of $\mathcal{ALCH}$ clauses and $\mathcal{N}^* = Res_{\mathcal{ALCH}}(\mathcal{N})$. The candidate model $\mathcal{I}$ is created based on $\mathcal{N}^*$ using the procedure presented in Section 4.2.3.

Denote by $\mathcal{N}^*_{\mathcal{ALC}}$ the set of all $\mathcal{ALC}$ clauses in $\mathcal{N}^*$, that is, all clauses which are not role inclusion axioms. Since $Res_{\mathcal{ALCH}}$ only adds rules to $Res_{\mathcal{ALC}}$, $Res_{\mathcal{ALC}}(\mathcal{N}^*_{\mathcal{ALC}}) = \mathcal{N}^*_{\mathcal{ALC}}$, that is, $\mathcal{N}^*_{\mathcal{ALC}}$ is saturated with respect to $Res_{\mathcal{ALC}}$. Hence, by Lemma 4.2.19, $\mathcal{I} \models C$ for every clause $C \in \mathcal{N}^*_{\mathcal{ALC}}$. All remaining clauses in $\mathcal{N}^* \setminus \mathcal{N}^*_{\mathcal{ALC}}$ are of the form $r \sqsubseteq s$. These clauses are satisfied in $\mathcal{I}$ by Lemma 5.1.2. Hence, $\mathcal{I} \models C$ for every clause $C \in \mathcal{N}^*$, and therefore $\mathcal{I}$ is a model of $\mathcal{N}^*$. This establishes refutational completeness of $\mathcal{N}^*$. □

Theorem 5.1.4. *$Int_{\mathcal{ALCH}}$ is interpolation complete for $\mathcal{ALCH}\nu$.*

Proof. Let $\mathcal{O}$ be an $\mathcal{ALCH}\nu$ ontology, $\mathcal{S}$ a signature, $\mathcal{N}^* = Int_{\mathcal{ALCH}}(Cl(\mathcal{O}))$ and $\mathcal{O}^{\mathcal{S}} = Ont(\mathcal{N}^*_{\mathcal{S}})$. If $Int_{\mathcal{ALCH}}$ is interpolation complete for $\mathcal{ALCH}\nu$, $\mathcal{O}^{\mathcal{S}}$ is a $\mathcal{ALCH}\nu$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$.

Note that the role hierarchy rule of $Int_{\mathcal{ALCH}}$ makes all entailments of the form $r \sqsubseteq s$ explicit in $\mathcal{N}^*$. Therefore, any those entailments in $\mathcal{S}$ are preserved by $\mathcal{O}^S$. That entailments of the form $C \sqsubseteq D$ and $C \equiv D$ are preserved as well can be shown in a similar fashion as for $Int_{\mathcal{ALC}}$.

In order to prove interpolation completeness of $Int_{\mathcal{ALC}}$ in Theorem 4.4.11, the following properties of $Res_{\mathcal{ALC}}$ and $Int_{\mathcal{ALC}}$ were exploited: (1) refutational completeness is preserved if the calculus only infers clauses with at most one negative definer literal (Theorem 4.3.3), (2) refutational completeness is preserved if we use ordered resolution (Theorem 4.3.5), and (3) whenever two definers D_1 and D_2 in $\mathcal{N} = Res_{\mathcal{ALC}}(\mathcal{O})$ are combined in $Res^{norm}_{\mathcal{ALC}}(\mathcal{N} \cup \mathcal{M})$ by a definer D, where $\mathcal{M} = Cl(\exists r^*.C)$, $r^* \notin sig(\mathcal{O})$ and $sig(C) \subseteq \mathcal{S}$, then D_1 and D_2 are combined by a definer D' in $\mathcal{N}^*$, and either $D' = D$, or $Res_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$ contains a clause $\neg D'' \sqcup D'$, and D'' occurs in a subsuming context to D in $Res_{\mathcal{ALC}}(\mathcal{N}^* \cup \mathcal{M})$ (Lemma 4.4.6). If $Int_{\mathcal{ALCH}}$ enjoys the same properties, it follows that we can adapt the proofs to show that $\mathcal{O}^{\mathcal{S}}$ preserves all entailments of the form $C \sqsubseteq D$ with $sig(C \sqsubseteq D) \subseteq \mathcal{S}$.

For (1), we note that in $Res_{\mathcal{ALCH}}$ and $Int_{\mathcal{ALCH}}$, nothing changes in the way definers are used and introduced. For (2), we observe that the model construction used to prove refutational completeness for $Res_{\mathcal{ALCH}}$ does not rely on a different ordering as the model construction for $Res_{\mathcal{ALC}}$. We can therefore refine $Res_{\mathcal{ALCH}}$ in the same way as $Res_{\mathcal{ALC}}$, and obtain a calculus $Res^{norm\prec_l}_{\mathcal{ALCH}}$ that is refutational complete. (3) regards definers that are introduced when $\mathcal{N} \cup \mathcal{M}$ is saturated. In order to be interpolation complete,

inferences of the empty clause from $\mathcal{N} \cup \mathcal{M}$ must have corresponding inferences of the empty clause from $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$. For this, it is crucial that every pair of definers from $\mathcal{N}$ that is combined in $\mathcal{N} \cup \mathcal{M}$ is combined in $\mathcal{N}^*$ by a corresponding definer. Which definers are combined in a clause set is determined by which role propagation rules can be applied on that clause set. Because of role hierarchies, in $\mathcal{ALCH}$ role restrictions with different roles can interact. The monotonicity rules make sure that none of these inferences get lost when clauses with role symbols outside $\mathcal{S}$ are removed from $\mathcal{N}^*$. We consider both ways in which role restrictions with different role symbols can interact.

Suppose $\mathcal{N}^* = \mathit{Int}_{\mathcal{ALCH}}(\mathcal{N})$ contains a clause of the form $C_1 = C_1' \sqcup \exists r.D_1$ and $\mathcal{M}$ contains a clause of the form $C_2 = C_2' \sqcup \forall s.D_2$. If $r \sqsubseteq_{\mathcal{N}} s$, $C_1' \sqcup \exists s.D_1 \in \mathcal{N}^*$ due to the $\exists$-monotonicity rule, and we can apply $\forall\exists$-role propagation on this clause and C_2. Assume $r \notin \mathcal{S}$. Then $C_1' \sqcup \exists s.D_1$ is still in $\mathcal{N}^{\mathcal{S}}$ due to the $\exists$-monotonicity rule, and the same inference is possible on $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$.

On the other hand, suppose $\mathcal{N}^*$ contains a clause of the form $C_1 = C_1' \sqcup \forall r.D_1$ and $\mathcal{M}$ contains a clause of the form $C_2 = C_2' \sqcup \exists s.D_2$ where $s \sqsubseteq_{\mathcal{N}} r$. Then, the $\exists$-monotonicity rule of $\mathit{Res}_{\mathcal{ALCH}}$ applies to C_2 and we can obtain the clause $C_2' \sqcup \exists r.D_1$ such that the $\forall\exists$-role propagation rule can be applied on this clause and C_1. Assume $r \notin \mathcal{S}$. Then, due to the $\forall$-monotonicity rule of $\mathit{Int}_{\mathcal{ALCH}}$, the clause $C_1' \sqcup \forall s.D_2$ is included in $\mathcal{N}^{\mathcal{S}}$, and we can make a corresponding $\forall\exists$-role propagation inference.

We see that $\mathit{Int}_{\mathcal{ALCH}}$ also enjoys property (3). Therefore, $\mathit{Res}_{\mathcal{ALCH}}$ and $\mathit{Int}_{\mathcal{ALCH}}$ enjoy all properties that are needed for showing interpolation completeness, and $\mathit{Int}_{\mathcal{ALCH}}$ is interpolation complete for $\mathcal{ALCH}\nu$. □

5.1.2 $\mathcal{ALCH}$ with Redundancy Elimination

The redundancy elimination rules in $\mathit{Res}^s_{\mathcal{ALC}}$ exploit the fact that a subsumption hierarchy is implied by clauses of the form $\neg D_1 \sqcup D_2$. This is used to define subsumption between literals of the form $\exists R.D$ or $\forall R.D$. This idea can be taken further in the presence of role inclusion axioms. Since $R \sqsubseteq S \models \exists R.D \sqsubseteq \exists S.D$ and $R \sqsubseteq S \models \forall S.D \sqsubseteq \forall R.D$, we can extend the redundancy notion defined by Definition 4.3.6.

Definition 5.1.5. A literal L_1 *subsumes a literal* L_2 *with respect to a clause set* $\mathcal{N}$, in symbols $L_1 \sqsubseteq_l L_2$, if one of the following conditions hold:

∀∃-Role Propagation

$$\frac{C_1 \sqcup \forall S.D_1 \qquad C_2 \sqcup \exists R.D_2 \qquad R \sqsubseteq_{\mathcal{N}} S}{C_1 \sqcup C_2 \sqcup \exists R.D_{12}}$$

where D_{12} is a possibly new definer representing $D_1 \sqcap D_2$.

∀∀-Role Propagation

$$\frac{C_1 \sqcup \forall R_1.D_1 \qquad C_2 \sqcup \forall R_2.D_2 \qquad R \sqsubseteq_{\mathcal{N}} R_1 \qquad R \sqsubseteq_{\mathcal{N}} R_2}{C_1 \sqcup C_2 \sqcup \forall R.D_{12}}$$

where D_{12} is a possibly new definer representing $D_1 \sqcap D_2$.

Figure 5.2: Modified role propagation rules for $Res^s_{\mathcal{ALCH}}$.

1. $L_1 = L_2$.
2. $L_1 = \exists R.D_1$, $L_2 = \exists S.D_2$, $R \sqsubseteq_{\mathcal{N}} S$ and either $D_1 = D_2$ or $\neg D_1 \sqcup D_2 \in \mathcal{N}$.
3. $L_1 = \forall R.D_1$, $L_2 = \forall S.D_2$, $S \sqsubseteq_{\mathcal{N}} R$ and either $D_1 = D_2$ or $\neg D_1 \sqcup D_2 \in \mathcal{N}$.

A clause C_1 *subsumes a clause* C_2 ($C_1 \sqsubseteq_c C_2$) if every literal $L_1 \in C_1$ subsumes a literal $L_2 \in C_2$. A clause C is *redundant* with respect to a clause set $\mathcal{N}$ if $\mathcal{N}$ contains a clause C' with $C' \sqsubseteq_c C$. The *reduction* of a clause C, *red*(C), is obtained from C by removing every literal that subsumes another literal in C.

With this notion of subsumption, conclusions of the monotonicity rules are redundant with respect to the premises. In order to preserve completeness, we integrate the monotonicity rules into the role propagation rules, which leads to the modified set of rules shown in Figure 5.2. $Res^s_{\mathcal{ALCH}}$ uses the modified ∀∃-role propagation rule, and the redundancy elimination rules introduced in Section 4.3, where the extended definitions in Definition 5.1.5 are used.

Theorem 5.1.6. *$Res^s_{\mathcal{ALCH}}$ is sound, terminating and refutationally complete.*

Proof. We can directly reuse the model construction used to show refutational completeness of $Res_{\mathcal{ALC}}$ without adapting the underlying ordering.

Suppose $\mathcal{N}^* = Res^s_{\mathcal{ALCH}}(\mathcal{N})$ is the saturation of any set of $\mathcal{ALCH}$ clauses using $Res^s_{\mathcal{ALCH}}$ after removing redundant clauses, such that $\bot \notin \mathcal{N}^*$. As observed earlier, $\mathcal{N}^*$ does not contain inferences of the ∃-monotonicity rule, since conclusions of

this rule are subsumed by the premise. However, we can saturate $\mathcal{N}^*$ using the $\exists$-monotonicity rule. Denote the resulting set by $\mathcal{N}^*_{mon}$.

In $\mathcal{N}^*_{mon}$, there is a literal $C \sqcup \exists S.D$ for every clause $C \sqcup \exists R.D$ with $R \sqsubseteq_{\mathcal{N}} S$. The model construction used to prove Theorem 4.2.20 can directly be applied on the set of clauses $\mathcal{N}^*_{mon} \setminus \{r \sqsubseteq s \in \mathcal{N}^*\}$ to build a candidate model $\mathcal{I}$ for $\mathcal{N}^*$. For the same reason as for $Res_{\mathcal{ALC}}$ and $Res_{\mathcal{ALCH}}$, $\mathcal{I}$ is a model of $\mathcal{N}^*$.

Due to Lemma 5.1.2 and Theorem 5.1.3, this implies that $Res^s_{\mathcal{ALCH}}$ is sound, terminating and refutationally complete. □

We can directly use the modified role propagation rules in $Int_{\mathcal{ALCH}}$, since they only integrate rules already present in the calculus. Note that we can use the same technique as is defined in Definition 4.4.12 to compute a minimal clausal representation $Int_{\mathcal{ALCH}}(Cl(\mathcal{O}), \mathcal{S})_{min}$ of the $\mathcal{ALCH}$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$. Required inferences of the monotonicity rules are still included in the minimal clausal representation of the uniform interpolant, since we only check redundancy against the resulting clause set. This means, inferences made by the monotonicity rules can be included in the uniform interpolant, even if they are redundant with respect to their premises.

In the following sections, we integrate the monotonicity rules into all rules where necessary.

5.2 Transitive Roles

The description logic $\mathcal{SH}$ extends $\mathcal{ALCH}$ with transitivity axioms of the form $\mathsf{trans}(r)$, which are part of the RBox of an $\mathcal{SH}$ ontology.

There are various known techniques to extend reasoning methods to description logics with transitivity axioms. While tableau-reasoning allows for a direct way to integrate transitivity into the calculus (see for example Halpern and Moses, 1992; Baader and Sattler, 2001), another common technique is to rewrite the input TBox in such a way that it can be processed by a reasoner that does not support transitive roles (Tobies, 2001; Schmidt and Hustadt, 2003a; Hustadt et al., 2004a; Carral et al., 2014). Our approach is to extend our calculus by a new rule. This rule has similarities to rewriting rules that have been used before. For example, the transformation presented in Schmidt and Hustadt (2003a) introduces similar clauses as our rule.

We first motivate the limitations of our method. The following example indicates that $\mathcal{SH}\nu$ is in general not expressive enough to represent uniform interpolants of $\mathcal{SH}$ ontologies for all signatures.

Example 5.2.1. Consider the following $\mathcal{SH}$ ontology $\mathcal{O}$:

$$s \sqsubseteq r$$
$$r \sqsubseteq t$$
$$\mathsf{trans}(r)$$

$\mathcal{O}$ has an infinite set of entailments of the following form, where C is any concept:

$$\mathcal{O} \models \forall t.C \sqsubseteq \forall s.C$$
$$\mathcal{O} \models \forall t.C \sqsubseteq \forall s.\forall s.C$$
$$\mathcal{O} \models \forall t.C \sqsubseteq \forall s.\forall s.\forall s.C$$
$$\vdots$$

Since neither t nor s are transitive, it is not clear how a finite $\mathcal{SHQ}\nu$ ontology in $sig(\mathcal{O}) \setminus \{r\}$ can capture these entailments.

In contrast, if the signature $\mathcal{S}$ to interpolate for contains all transitive roles of the ontology $\mathcal{O}$, there is always a finite $\mathcal{SH}\nu$ uniform interpolant of $\mathcal{O}$. Therefore, the calculus we present in this section is only interpolation complete for forgetting intransitive roles.

Since there is only one new type of axioms, the extension of the normal form is straightforward.

Definition 5.2.2. An $\mathcal{SH}$ *clause* is either an $\mathcal{ALCH}$ clause or is of the form $\mathsf{trans}(r)$, where $r \in N_r$. An ontology $\mathcal{N}$ is in $\mathcal{SH}$ normal form if every axiom in $\mathcal{N}$ is an $\mathcal{SH}$ clause.

The calculi $Res_{\mathcal{SH}}$ and $Int_{\mathcal{SH}}$ extend $Res_{\mathcal{ALCH}}$ and $Int_{\mathcal{ALCH}}$ by the transitivity rule shown in Figure 5.3, which introduces a new definer D'. For termination, this rule is only applied if no corresponding definer has been introduced yet.

The transitivity rule is motivated as follows. Assume a domain element x of any model satisfies $\forall r.D$, where r has a transitive sub-role s. Since s is transitive, every

Transitivity Rule

$$\frac{C \sqcup \forall R.D \qquad \mathsf{trans}(S) \qquad S \sqsubseteq_{\mathcal{N}} R}{C \sqcup \forall S.D' \quad \neg D' \sqcup D \quad \neg D' \sqcup \forall S.D'}$$

Figure 5.3: Transitivity rule of $Res_{\mathcal{SH}}$ and $Int_{\mathcal{SH}}$.

element x' that is connected to x by a chain of s-successors is also an s-successor of x. Since s is a sub-role of r, every such x' is also an r-successor of x, and has to satisfy D as well. We express this information by introducing a new cyclic definer D' such that $D' \sqsubseteq D$. In interaction with the $\forall\exists$-role propagation rule, this ensures that every s-successor satisfying D' has only s-successors that also satisfy D'. Because $D' \sqsubseteq D$, all these s-successors also satisfy D. This way we express on a clausal level that if a domain element x satisfies $\forall r.D$, every element x' that is connected to x by a chain of s-successors satisfies D as well.

Example 5.2.3. Consider the following ontology $\mathcal{O}_{\mathcal{SH}}$.

$$A \sqsubseteq \forall s.(A \sqcup B)$$
$$A \sqsubseteq \exists r.\exists r.\neg B$$
$$r \sqsubseteq s$$
$$\mathsf{trans}(r)$$

We want to forget B, and compute a uniform interpolant for $\mathcal{S}_{\mathcal{SH}} = \{A, r, s\}$. The $\mathcal{SH}$ normal form of $\mathcal{O}_{\mathcal{SH}}$ consists of the following clauses:

1. $\neg A \sqcup \forall s.D_1$

2. $\neg D_1 \sqcup A \sqcup B$

3. $\neg A \sqcup \exists r.D_2$

4. $\neg D_2 \sqcup \exists r.D_3$

5. $\neg D_3 \sqcup \neg B$

6. $r \sqsubseteq s$

7. $\mathsf{trans}(r)$

The transitivity rule is applied on Clause 1 and the two RBox axioms.

8. $\neg A \sqcup \forall r.D'_1$ *(Transitivity 1,6,7)*
9. $\neg D'_1 \sqcup D_1$ *(Transitivity 1,6,7)*
10. $\neg D'_1 \sqcup \forall r.D'_1$ *(Transitivity 1,6,7)*
11. $\neg D'_1 \sqcup A \sqcup B$ *(Resolution 2, 9)*

In order to derive clauses that allow for inferences on B, several applications of the $\forall\exists$-role propagation rule have to be performed.

12. $\neg A \sqcup \exists r.D'_{12}$ *($\forall\exists$-role propagation 3, 8)*
13. $\neg D'_{12} \sqcup D'_1$
14. $\neg D'_{12} \sqcup D_2$
15. $\neg D'_{12} \sqcup \exists r.D_3$ *(Resolution 4, 14)*
16. $\neg D'_{12} \sqcup D_1$ *(Resolution 9, 13)*
17. $\neg D'_{12} \sqcup \forall r.D'_1$ *(Resolution 10, 13)*
18. $\neg D'_{12} \sqcup A \sqcup B$ *(Resolution 11, 13)*
19. $\neg D'_{12} \sqcup \exists r.D'_{13}$ *($\forall\exists$-role propagation 15, 17)*
20. $\neg D'_{13} \sqcup D'_1$
21. $\neg D'_{13} \sqcup D_3$
~~22.~~ $\neg D'_{13} \sqcup A \sqcup B$ *(Resolution 11, 20)*
23. $\neg D'_{13} \sqcup \neg B$ *(Resolution 5, 21)*
24. $\neg D'_{13} \sqcup A$ *(Resolution 22, 23)*

More rule applications are possible, but all further inference steps infer redundant clauses and clauses that are not necessary for the uniform interpolant. The clauses with a bold face number are in the clausal form of the uniform interpolant. Note that due to Theorem 4.4.13, we can omit clauses that are not the direct result of resolution on B, or do not contain a definer that has to be included in the clausal representation of the uniform interpolant. In particular, we can omit the conclusions of the transitivity rule (Clauses 8–10), since the definer D' does not have to be included in the uniform interpolant.

After eliminating all definers, we obtain the following uniform interpolant $\mathcal{O}^{\mathcal{S}}_{\mathcal{SH}}$.

$$A \sqsubseteq \exists r.\exists r.A \qquad r \sqsubseteq s \qquad \mathsf{trans}(r)$$

In order to prove refutational completeness of $Res^{s}_{\mathcal{SH}}$, we adapt the candidate model construction to incorporate transitive roles. Let $\mathcal{N}$ be any set of $\mathcal{SH}$ clauses, $\mathcal{N}^* = Res^{s}_{\mathcal{SH}}(\mathcal{N})$ be the saturation of $\mathcal{N}$, and suppose $\bot \notin \mathcal{N}^*$. Following the construction presented in Section 4.2.3, together with the modifications for role hierarchies discussed in Section 5.1.2, we build the interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ based on the clauses in $\mathcal{N}^*$. In order to prove the refutational completeness of $Res_{\mathcal{SH}}$, $\mathcal{I}$ has to be extended to incorporate transitivity axioms. $\mathcal{I}_{\mathsf{trans}} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}_{\mathsf{trans}}} \rangle$ is defined as follows, where r^* denotes the transitive closure of r.

1. For every $A \in N_c$, $A^{\mathcal{I}_{\mathsf{trans}}} = A^{\mathcal{I}}$.
2. For every $r \in N_r$, $r^{\mathcal{I}_{\mathsf{trans}}} = r^{\mathcal{I}} \cup \{(x, x') \mid (x, x') \in (s^{\mathcal{I}})^*, s \sqsubseteq_{\mathcal{N}} r, \mathsf{trans}(s) \in \mathcal{N}\}$.

This modification ensures (i) that $r^{\mathcal{I}_{\mathsf{trans}}}$ is transitive if $\mathsf{trans}(r) \in \mathcal{N}$ and (ii) that for all roles with $s \sqsubseteq_{\mathcal{N}} r$, we still have $s^{\mathcal{I}_{\mathsf{trans}}} \subseteq r^{\mathcal{I}_{\mathsf{trans}}}$. This way, it ensures that $\mathcal{I} \models \mathsf{trans}(r)$ for all $\mathsf{trans}(r) \in \mathcal{N}^*$, and $\mathcal{I} \models r \sqsubseteq s$ for all $r \sqsubseteq s \in \mathcal{N}^*$.

Lemma 5.2.4. *For every clause $C \in \mathcal{N}^*$, $\mathcal{I}_{\mathsf{trans}} \models C$.*

Proof. We already observed that $\mathcal{I}_{\mathsf{trans}} \models C$ for all clauses C of the form $r \sqsubseteq s$ and $\mathsf{trans}(r)$. Note that the satisfaction of concept symbols and existential restrictions is unaffected by the modification, since we only add additional edges between individuals. Therefore, if the maximal literal in C is of the form A, $\neg A$ or $\exists r.D$, $\mathcal{I}_{\mathsf{trans}} \models C$ follows from the results in Lemma 4.2.19 and Lemma 5.1.2, which show that the model construction for $Res_{\mathcal{ALC}}$ and $Res_{\mathcal{ALCH}}$ always succeeds. $\mathcal{I}_{\mathsf{trans}} \models C$ also if the maximal literal in C is of the form $\forall r.D$ and there is no role s with $\mathsf{trans}(s) \in \mathcal{N}$ and $s \sqsubseteq_{\mathcal{N}} r$, since in this case $r^{\mathcal{I}_{\mathsf{trans}}} = r^{\mathcal{I}}$.

The only remaining case is that the maximal literal in C is of the form $\forall r.D$ and there is a role s with $\mathsf{trans}(s) \in \mathcal{N}$ and $s \sqsubseteq_{\mathcal{N}} r$. We prove this case by contradiction. Assume C is the smallest clause in $\mathcal{N}^*$ with $\mathcal{I}_{\mathsf{trans}} \not\models C$, the maximal literal in C is of the form $\forall r.D$ and there is a role s with $\mathsf{trans}(s) \in \mathcal{N}$ and $s \sqsubseteq_{\mathcal{N}} r$. Assume $C = C' \sqcup \forall r.D$. Since $\mathcal{I}_{\mathsf{trans}} \not\models C$, there is a domain element $x_{D_1} \in \Delta^{\mathcal{I}_{\mathsf{trans}}}$

with $x_{D_1} \notin C^{\mathcal{I}_{\mathsf{trans}}}$. This implies $x_{D_1} \notin (\forall r.D)^{\mathcal{I}_{\mathsf{trans}}}$. Therefore, there must be a domain element x_{D_2} with $(x_{D_1}, x_{D_2}) \in r^{\mathcal{I}_{\mathsf{trans}}}$ and $x_{D_2} \notin D^{\mathcal{I}_{\mathsf{trans}}}$. We can establish that $(x_{D_1}, x_{D_2}) \notin r^{\mathcal{I}}$, since $(x_{D_1}, x_{D_2}) \in r^{\mathcal{I}}$ would imply $x_{D_2} \in D^{\mathcal{I}}$ by Lemma 4.2.19. We therefore have $(x_{D_1}, x_{D_2}) \in r^{\mathcal{I}_{\mathsf{trans}}} \setminus r^{\mathcal{I}}$. This implies $(x_{D_1}, x_{D_2}) \in (t^{\mathcal{I}})^*$ for some role t with $\mathsf{trans}(t) \in \mathcal{N}$ and $t \sqsubseteq_{\mathcal{N}} r$. Without loss of generality, we may assume that $t = s$. Since $(x_{D_1}, x_{D_2}) \in (s^{\mathcal{I}})^*$, there is a chain of s-edges in $\mathcal{I}$ that connect x_{D_1} with x_{D_2}. Due to our modification of the model construction in Section 5.1.2, and since $s \sqsubseteq_{\mathcal{N}} r$, we have an r-edge in $\mathcal{I}$ for every such s-edge of the chain.

Therefore, x_{D_1} and x_{D_2} are connected in $\mathcal{I}$ by a chain of r-edges along domain elements $x_{D'_1}, x_{D'_2}, \ldots, x_{D'_n}$, where $D'_1 = D_1$ and $D'_n = D_2$. An r-edge between two domain elements x_i and x_{i+1} is only added by the model construction if there is corresponding model fragment $I^{D'_i}$ with $\exists r.D'_{i+1} \in I^{D'_i}$ for $1 \leq i < n$. If $\exists r.D'_{i+1} \in I^{D'_i}$, there must be clauses $C_{j_i} = C'_{j_i} \sqcup \exists r.D'_{i+1}$, where $\exists r.D'_{i+1}$ is maximal and $I^{D'_i}_{j_i - 1} \not\models C_{j_i}$.

For the first domain element of the chain, x_{D_1}, we have $x_{D_1} \notin (C')^{\mathcal{I}}$. Therefore, $I^{D_1} \not\models C'$, since $C' = C \setminus \{\forall r.D\}$. We also have $I^{D_1} \not\models C'_{j_1}$, due to negative monotonicity of the model construction (Lemma 4.2.14). Observe that, due to the transitivity rule and the clause $C = C' \sqcup \forall r.D$, there are also the clauses $C_1^{\mathsf{trans}} = C' \sqcup \forall r.D'$, $C_2^{\mathsf{trans}} = \neg D' \sqcup D$ and $C_2^{\mathsf{trans}} = \neg D' \sqcup \forall r.D'$. $\forall\exists$-role propagation on C_{j_1} and C_1^{trans} produces the clause $C_k = C' \sqcup C'_{j_1} \sqcup \exists r.D_k$, with $\neg D_k \sqcup D' \in \mathcal{N}'$. We have $C_k = C_{j_1}$ and $D_k = D'_2$, since otherwise $C_k \prec_c C_{i_1}$, $\exists r.D_k \in I^{D_1}_{i_1 - 1}$ and $I^{D_1}_{i_1 - 1} \models C_{i_1}$. Therefore, $\neg D'_2 \sqcup D' \in \mathcal{N}^*$ and $\neg D'_2 \sqcup \forall r.D' \in \mathcal{N}^*$.

Now, by induction on i starting from 2, we can show that $\neg D'_i \sqcup D' \in \mathcal{N}^*$ and $\neg D'_i \sqcup \forall r.D' \in \mathcal{N}^*$ for all $2 \leq i \leq n$. Since $\mathcal{I} \models \neg D_{j_i} \sqcup \forall r.D'$ due to completeness of $Res_{\mathcal{ALCH}}$, and $D'_i \in I^{D'_i}$ due to the construction of the model fragments, $D' \in I^{D'_{i+1}}$. By Lemma 4.2.17, this implies $\neg D'_{i+1} \sqcup D' \in \mathcal{N}^*$. Resolution on D' and $\neg D' \sqcup \forall r.D'$ results in $\neg D'_{i+1} \sqcup \forall r.D' \in \mathcal{N}^*$.

For the last element of the chain, $D'_n = D_2$, and hence $\neg D_2 \sqcup D' \in \mathcal{N}^*$. Resolution on D' and $\neg D' \sqcup D$ gives us $\neg D_2 \sqcup D \in \mathcal{N}^*$, and therefore $D \in I^{D_2}$ and $x_{D_2} \in D^{\mathcal{I}}$ by the model construction. Since the interpretations of concept symbols are the same in $\mathcal{I}_{\mathsf{trans}}$, this implies $x_{D_2} \in D^{\mathcal{I}_{\mathsf{trans}}}$. But this contradicts the observation that there is a domain element x_{D_2} such that $(x_{D_1}, x_{D_2}) \in r^{\mathcal{I}_{\mathsf{trans}}}$ and $x_{D_2} \notin D^{\mathcal{I}_{\mathsf{trans}}}$, and the initial assumption that $x_{D_1} \notin (\forall r.D)^{\mathcal{I}_{\mathsf{trans}}}$.

We obtain that $\mathcal{I}_{\mathsf{trans}} \models C$ for all $C \in \mathcal{N}^*$, and that $\mathcal{I}_{\mathsf{trans}}$ is a model of $\mathcal{N}^*$. □

This lemma allows us to establish refutational completeness of $Res^s_{\mathcal{SH}}$.

Theorem 5.2.5. *$Res_{\mathcal{SH}}$ is terminating, sound and refutationally complete for $\mathcal{SH}$.*

To prove interpolation completeness of $Int_{\mathcal{SH}}$, there is not much to be done, since interpolation completeness is already proved for $Int_{\mathcal{ALCH}}$, and we cannot forget transitive roles with this calculus.

Theorem 5.2.6. *$Int_{\mathcal{SH}}$ is interpolation complete for forgetting intransitive roles and concepts in $\mathcal{SH}\nu$.*

Proof. Let $\mathcal{O}$ be any $\mathcal{SH}$ ontology and $\mathcal{S}$ be any signature such that for all $\mathsf{trans}(r) \in \mathcal{O}$ we have $r \in \mathcal{S}$. Let $\mathcal{N} = Cl(\mathcal{O})$ and $\mathcal{N}^{\mathcal{S}} = (Int_{\mathcal{SH}}(\mathcal{N})_n)_{\mathcal{S}}$. Let C be a concept such that $sig(C) \subseteq \mathcal{S}$ and $\mathcal{M} = Cl(\exists r^*.C)$, where $r^* \notin sig(\mathcal{O})$. We have to show that whenever $\mathcal{N} \cup \mathcal{M}$ is unsatisfiable, $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ is unsatisfiable as well (see proofs for Theorem 4.4.11 and Theorem 5.1.4).

The only new rule in $Res_{\mathcal{SH}}$ is the transitivity rule, which allows for inferences on transitivity axioms. Due to the restriction on the signature, all these axioms are still present in $\mathcal{N}^{\mathcal{S}}$, and the role hierarchy rule ensures that $\mathcal{N}^{\mathcal{S}}$ preserves all entailments of the form $r \sqsubseteq s$, $r, s \in \mathcal{S}$. Therefore, if the transitivity rule can be applied on any clause in $\mathcal{N} \cup \mathcal{M}$, a similar application is possible in $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$. □

5.3 Inverse Roles

$\mathcal{ALCHI}$ extends $\mathcal{ALCH}$ with roles of the form r^-, $r \in N_r$. Recall that given a role R, we denote by $\mathsf{Inv}(R)$ its inverse, that is $\mathsf{Inv}(r) = r^-$ and $\mathsf{Inv}(r^-) = r$. The normal form for $\mathcal{ALCHI}$ is the same as for $\mathcal{ALCH}$, only that we also allow roles of the form r^-.

Observe that the following $\mathcal{ALCHI}$ axioms are equivalent:

$$\top \sqsubseteq C_1 \sqcup \forall R.C_2$$

$$\top \sqsubseteq C_2 \sqcup \forall \mathsf{Inv}(R).C_1$$

This equivalence can be understood from the first-order logic representation of both axioms. If we represent C_1 and C_2 using two unary predicates, and R using

a binary predicate, both axioms are represented as $\forall x, y : C_1(x) \vee R(x, y) \vee C_2(y)$ (see Section 3.1). The equivalence cannot directly be expressed as a rule, since in general, C_1 and C_2 might be complex concepts, and complex concepts under a role restriction are not allowed in the normal form. Instead, we introduce further definers for the concept that would occur under the universal role restriction. In order to preserve termination of the calculus, this has to be done in a controlled way that avoids the introduction of arbitrarily many new definer symbols. This is achieved by using an additional normal form transformation step, which is applied directly after the normal form transformation introduced in Section 4.1.1. For this, the following transformation rule is used.

$$C \sqcup \forall R.D \Longrightarrow \neg D_{\forall R.D} \sqcup C, \quad D_{\forall R.D} \sqcup \forall R.D \tag{5.1}$$

The transformation is sound according to our definition of soundness, because it does not introduce any new entailments without definer symbols. The first clause is equivalent to the concept inclusion $D_{\forall R.D} \sqsubseteq C$, and the second axiom is of a form that allows us to make use of the equivalence observed at the beginning of this section, since the other than the universal restriction it only contains a concept symbol.

Using this transformation, we ensure that every universal restriction in the clause set occurs in a clause of the form $D_{\forall R.D} \sqcup \forall R.D$. The transformation is applied once for each universal restriction in each clause. For example, the clause $A \sqcup \forall r.D_1 \sqcup \forall r.D_2$ is transformed into the following set of clauses:

$$A \sqcup \forall r.D_2 \sqcup \neg D_{\forall r.D_1}, \quad D_{\forall r.D_1} \sqcup \forall r.D_1$$
$$A \sqcup \forall r.D_1 \sqcup \neg D_{\forall r.D_2}, \quad D_{\forall r.D_2} \sqcup \forall r.D_2$$

For the clause $A \sqcup \forall r.D_1 \sqcup \forall r.D_2 \sqcup \forall r.D_3$, the transformation produces three different clauses. Note that, in contrast to the standard normal form transformation, the output of this transformation may not be a normal clause set according to Definition 4.1.4, since clauses with more than one negative definer literal are possible. For instance, the clause $\neg D_1 \sqcup \forall r.D_2$ is transformed to $\neg D_1 \sqcup \neg D_{\forall r.D_2}$ and $D_{\forall r.D_2} \sqcup \forall r.D_2$.

For termination, it is crucial that the transformation is only applied once, as part of the normal form transformation, and not during the application of the calculus. This is why the transformation is used as part of the normal form transformation,

Role Inversion

$$\frac{D_1 \sqcup \forall R.D_2}{D_2 \sqcup \forall \mathsf{Inv}(R).D_1}$$

Figure 5.4: Role inversion rule of $Res_{\mathcal{ALCHI}}$ and $Int_{\mathcal{ALCHI}}$.

.

and not as part of the calculus. Note that the normal form transformation still only introduces linearly many definer symbols (one for each existential restriction and two for each universal restriction).

The calculi $Res_{\mathcal{ALCHI}}$ and $Int_{\mathcal{ALCHI}}$ work on any set of clauses transformed this way, and extend $Res^s_{\mathcal{ALCH}}$ and $Int_{\mathcal{ALCH}}$ by the role inversion rule shown in Figure 5.4. This rule directly implements the equivalence we showed at the beginning of the section, while preserving the normal form in its conclusion, thanks to the additionally introduced definers. Since the number of definers after the normal form transformation is still linear in the input size, the number of introduced definer symbols is still bounded by $O(2^n)$.

Example 5.3.1. Consider the following ontology $\mathcal{O}_{\mathcal{ALCHI}}$:

$$A_0 \sqsubseteq \exists r.B_0$$
$$B_0 \sqsubseteq \exists r.B_1 \sqcap (A_1 \sqcup B_2)$$
$$B_1 \sqsubseteq \forall r^-.\neg B_2$$

One can verify that $\mathcal{O}_{\mathcal{ALCHI}} \models A_0 \sqsubseteq \exists r.A_1$. This is visualised in Figure 5.5. Every instance of A_0 has an r-successor satisfying B_0, and every instance of B_0 has an r-successor satisfying B_1. Additionally, every B_0 instance satisfies $A_1 \sqcup B_2$, due to the second axiom in the ontology. Because of the third axiom of the ontology, every instance of B_1 has only r-predecessors that do not satisfy B_2. As we can see in the graph, every instance of B_0 is an r-predecessor of an instance of B_1. Hence, every instance of B_0 satisfies $\neg B_2$, and by resolution with $A_1 \sqcup B_2$, we can conclude that $B_0 \sqsubseteq A_1$ and $A_0 \sqsubseteq \exists r.A_1$.

In order to verify that our calculus infers the same conclusion, we compute the uniform interpolant for $\mathcal{S}_{\mathcal{ALCHI}} = \{A_0, A_1, r\}$. The initial normal form transformation

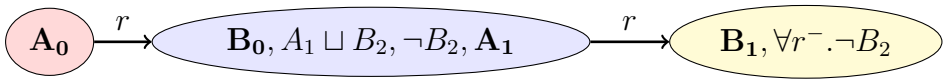

Figure 5.5: Model of $\mathcal{O}_{\mathcal{ALCHI}}$, an ontology with inverse roles.

produces the following clause set:

1. $\neg A_0 \sqcup \exists r.D_1$

2. $\neg D_1 \sqcup B_0$

3. $\neg B_0 \sqcup \exists r.D_2$

4. $\neg D_2 \sqcup B_1$

5. $\neg B_0 \sqcup A_1 \sqcup B_2$

6. $\neg B_1 \sqcup \forall r^-.D_3$

7. $\neg D_3 \sqcup \neg B_2$

Clause 6 contains a universal role restriction. Hence, for $\mathcal{ALCHI}$, we finish the normal form transformation by adding the following clauses:

8. $\neg D_{\forall r^-.D_3} \sqcup \neg B_1$

9. $D_{\forall r^-.D_3} \sqcup \forall r^-.D_3$

We apply $Int_{\mathcal{ALCHI}}$ to the clause set. Because we want to compute the uniform interpolant for $\mathcal{S}_{\mathcal{ALCHI}} = \{A_0, A_1, r\}$, we have to infer all resolvents on B_0, B_1 and B_2, as well the transitive closure under definers. Clause 6 remains in the clause set as resolvent of Clause 8 and 9. Note that inferred clauses with more than one negative definer are not needed. We process the concepts B_0, B_1 and B_2 one after the other.

10. $\neg D_1 \sqcup \exists r.D_2$ *(Resolution 2, 3 on B_0)*

11. $\neg D_1 \sqcup A_1 \sqcup B_2$ *(Resolution 2, 5 on B_0)*

12. $\neg D_2 \sqcup \forall r^-.D_3$ *(Resolution 4, 6 on B_1)*

13. $D_3 \sqcup \forall r.D_{\forall r^-.D_3}$ *(Role Inversion 9)*

~~14~~. $\neg D_1 \sqcup D_3 \sqcup \exists r.D_{2\forall r^-.D_3}$ *(∀∃-Role Propagation 10, 13)*

~~15~~. $\neg D_{2\forall r^-.D_3} \sqcup D_2$ *($D_{2\forall r^-.D_3} \sqsubseteq D_2$)*

~~16~~. $\neg D_{2\forall r^-.D_3} \sqcup D_{\forall r^-.D_3}$ *($D_{2\forall r^-.D_3} \sqsubseteq D_{\forall r^-.D_3}$)*

~~17~~. $\neg D_{2\forall r^-.D_3} \sqcup B_1$ *(Resolution 4, 15 on D_2)*

~~18~~. $\neg D_{2\forall r^-.D_3} \sqcup \neg B_1$ *(Resolution 8, 16 on D')*

19. $\neg D_{2\forall r^-.D_3}$ *(Resolution 17, 18 on B_1)*

20. $\neg D_1 \sqcup D_3$ *($\exists$-Elimination 14, 19)*

21. $\neg D_1 \sqcup \neg B_2$ *(Resolution 7, 20 on D_3)*

22. $\neg D_1 \sqcup A_1$ *(Resolution 11, 21 on B_2)*

All further inferences are redundant or produce clauses that are not needed for the uniform interpolant. The clauses with a bold face number form the clausal form representation of the uniform interpolant. After eliminating all definers, we obtain the following ontology, which is a uniform interpolant of $\mathcal{O}_{\mathcal{ALCHI}}$ for $\mathcal{S}_{\mathcal{ALCHI}}$:

$$A_0 \sqsubseteq \exists r.(\exists r.\forall r^-.\top \sqcap A_1)$$

Because $\forall r^-.\top$ is tautological, the uniform interpolant can be simplified to the following uniform interpolant $\mathcal{O}^{\mathcal{S}}_{\mathcal{ALCHI}}$:

$$A_0 \sqsubseteq \exists r.(\exists r.\top \sqcap A_1)$$

We see that $\mathcal{O}^{\mathcal{S}}_{\mathcal{ALCHI}} \models A_0 \sqsubseteq \exists r.A_1$, as observed in the beginning of the example.

Before we prove to the completeness properties of $Res^{s}_{\mathcal{ALCHI}}$ and $Int_{\mathcal{ALCHI}}$, we prove the following lemma about sets saturated using $Res_{\mathcal{ALCHI}}$.

Lemma 5.3.2. *Let $\mathcal{N}$ be any set of clauses transformed using the above transformations, and let $\forall R.D$ be any universal restriction occurring in $\mathcal{N}^* = Res_{\mathcal{ALCHI}}(\mathcal{N})$. Then, there is a clause $D_{\forall R.D} \sqcup \forall R.D \in \mathcal{N}^*$ assigning a unique definer $D_{\forall R.D}$ to $\forall R.D$, such that for every clause $C \sqcup \forall R.D \in \mathcal{N}^*$, either $C = D_{\forall R.D}$ or there is also the clause $C \sqcup \neg D_{\forall R.D} \in \mathcal{N}^*$.*

Proof. Let $\mathcal{N}^*$ be as in the lemma. In $Res_{\mathcal{ALCHI}}$, the only rule that introduces universal restrictions that are not in the initial clause set is the role inversion rule. (Note that the $\forall\forall$-role propagation rule is only part of $Int_{\mathcal{ALCHI}}$, not of $Res_{\mathcal{ALCHI}}$.) Let $\forall R.D$ be a universal role restriction occurring in $\mathcal{N}^*$. We distinguish two cases based on why $\forall R.D$ occurs in $\mathcal{N}^*$.

1. $\forall R.D$ occurs in the initial clause set $\mathcal{N}$ in the clause $C \sqcup \forall R.D$. Then, the normal form transformation adds the two clauses $D_{\forall R.D} \sqcup \forall R.D$ and $\neg D_{\forall R.D} \sqcup C$. Let ρ

be a sequence of inferences involving $C \sqcup \forall R.D$ and resulting in another clause $C' \sqcup \forall R.D$. ρ can only involve inferences on literals in C. A similar sequence of inferences is possible starting from the clause $\neg D_{\forall R.D} \sqcup C$, and resulting in the clause $C' \sqcup \neg D_{\forall R.D} \in \mathcal{N}^*$.

2. $\forall R.D$ occurs in a clause $C \sqcup \forall R.D \in \mathcal{N}^*$ that is the result of the role inversion rule. Then $C = D_{\forall R.D}$, and the premise of the role inversion rule is $D \sqcup \forall \mathsf{Inv}(R).D_{\forall R.D}$. $D_{\forall R.D}$ is a definer introduced by the initial normal form transformation, which also introduces a clause of the form $\neg D_{\forall R.D} \sqcup C_1$. Resolution between $D_{\forall R.D} \sqcup \forall R.D$ and $\neg D_{\forall R.D} \sqcup C_1$ results in the clause $C_1 \sqcup \forall R.D$. Therefore, we have the two clauses $C_1 \sqcup \forall R.D$ and $D_{\forall R.D} \sqcup \forall R.D$, and we can argue in the same way as for Point 1 that for every other clause $C_2 \sqcup \forall R.D \in \mathcal{N}^*$, we also have the clause $C_2 \sqcup \neg D_{\forall R.D} \in \mathcal{N}^*$.

We have established that regardless of why $\forall R.D$ occurs in $\mathcal{N}^*$, there is always a corresponding definer $D_{\forall R.D}$, such that every clause $C \sqcup \forall R.D \in \mathcal{N}^*$ has a corresponding clause $\neg D_{\forall R.D} \sqcup C \in \mathcal{N}^*$. □

In order to prove the refutational completeness of $Res^s_{\mathcal{ALCHI}}$, we consider a set of clauses $\mathcal{N}$ and its saturation $\mathcal{N}^* = Res^s_{\mathcal{ALCHI}}(\mathcal{N})$ such that $\bot \notin \mathcal{N}^*$, and describe how we can build a model for $\mathcal{N}^*$.

As in the refutational completeness proof of $Res^s_{\mathcal{ALCH}}$, we first saturate $\mathcal{N}^*$ using the $\exists$-monotonicity rule (see Theorem 5.1.3). This step is necessary because conclusions of the $\exists$-monotonicity rule are redundant according to our definition of redundancy (see Definition 5.1.5), but enable us to build an interpretation that satisfies all role inclusions. Then we build for each definer D occurring in $\mathcal{N}^*$ a model fragment I^D using the construction for $Res_{\mathcal{ALC}}$ presented in Section 4.2.3, where we treat roles of the form r^- as role symbols.

Since this time, literals of the form $\exists r^-.D$ can occur in the model fragments, we have to adapt the construction of the candidate model $\mathcal{I}$. The candidate model is now defined as follows.

- For every atomic concept A, $A^{\mathcal{I}} = \{x_D \mid A \in I^D\}$.
- For every role r, $r^{\mathcal{I}} = \{(x_{D_1}, x_{D_2}) \mid \exists r.D_2 \in I^{D_1}\} \cup \{(x_{D_2}, x_{D_1}) \mid \exists r^-.D_2 \in I^{D_1}\}$.

In candidate models for $Res_{\mathcal{ALC}}$, all r-edges from a domain element x_{D_1} to another domain element x_{D_2} are solely determined by the model fragment I^{D_1}. For the candidate models defined here, the situation is different, since both literals in I^{D_1} and I^{D_2} can contribute to an R-edge from x_{D_1} to x_{D_2}. This affects the satisfiability of universal restrictions, which cannot be solely determined on the basis of model fragments any more. Instead, two model fragments have to be taken into account to decide whether a literal of the form $\forall R.D$ is satisfied by a domain element. For all remaining literals, the situation is the same as for $Res_{\mathcal{ALC}}$, and we do not have to reconsider these cases.

Lemma 5.3.3. *$\mathcal{I}$ is a model for $\mathcal{N}^*$.*

Proof. First observe that all clauses of the form $R \sqsubseteq S \in \mathcal{N}^*$ are satisfied by the candidate model for the same reasons as for candidate models for $Res_{\mathcal{ALCH}}$ and $Res^s_{\mathcal{ALCH}}$ (see Lemma 5.1.3 and Theorem 5.1.6). For the remaining clauses, we do the proof by contradiction.

Assume C_i is the smallest clause for which $\mathcal{I} \not\models C_i$. We only consider the maximal literal L in C_i and set $C_i = C_i' \sqcup L$. L cannot be of the form A, $\neg A$ or $\exists R.D_1$, since then $\mathcal{I} \not\models L$ leads to a contradiction for the same reasons as for $Res_{\mathcal{ALCH}}$.

Assume $L = \forall R.D_1$. Since $\mathcal{I} \not\models C_i$, also $\mathcal{I} \not\models \forall R.D_1$, and there must be an individual $x_D \in \Delta^{\mathcal{I}}$ such that $x_D \notin (C_i')^{\mathcal{I}}$ and $x_D \notin (\forall r.D_1)^{\mathcal{I}}$. Since $x_D \notin (\forall r.D_1)^{\mathcal{I}}$, there is a domain element $x_{D_2} \in \Delta^{\mathcal{I}}$ such that $(x_D, x_{D_2}) \in R^{\mathcal{I}}$ and $x_{D_2} \notin D_1^{\mathcal{I}}$. The model construction adds an edge (x_D, x_{D_2}) to $R^{\mathcal{I}}$ if $\exists R.D_2 \in I^D$ or $\exists \mathsf{Inv}(R).D \in I^{D_2}$. We consider both cases.

1. $\exists R.D_2 \in I^D$. Then both $\exists R.D_2$ and $\forall R.D_1$ are maximal in some clauses in $\mathcal{N}^*$, and a contradiction arises for the same reasons as for $Res_{\mathcal{ALC}}$. Observe that from this it follows that, whenever $\forall Q.D''$ is maximal in a clause $C'' \sqcup \forall Q.D''$, and there is an edge $(x, x_{D'}) \in Q^{\mathcal{I}}$ due to a literal $\exists Q.D' \in I$, then either $x \in (C'')^{\mathcal{I}}$ or $x_{D'} \in (D'')^{\mathcal{I}}$.

2. $\exists \mathsf{Inv}(R).D \in I^{D_2}$. Then, there is a clause $C_j = C_j' \sqcup \exists \mathsf{Inv}(R).D$ in which $\exists \mathsf{Inv}(R).D$ is maximal, and $I^{D_2}_{j-1} \not\models C_j$.

   Due to Lemma 5.3.2, since $C_i = C_i' \sqcup \forall R.D_1$, either $C_i' = D_{\forall R.D_1}$, or we also have the two clauses $D_{\forall R.D_1} \sqcup \forall R.D_1 \in \mathcal{N}^*$ and $C_i' \sqcup \neg D_{\forall R.D_1} \in \mathcal{N}^*$. We show

that in both cases, $x_D \notin (D_{\forall R.D_1})^{\mathcal{I}}$. If $C_i' = D_{\forall R.D_1}$, since $x_D \notin (C_i')^{\mathcal{I}}$, we have $x_D \notin (D_{\forall R.D_1})^{\mathcal{I}}$. If $C_i' \neq D_{\forall R.D_1}$, observe that $\neg D_{\forall R.D_1} \prec_l \forall R.D_1$, and that $\forall R.D_1$ is maximal in C_i. Therefore, $C_i' \sqcup \neg D_{\forall R.D_1} \prec_c C_i$, and $C_i' \sqcup \neg D_{\forall R.D_1}$ is satisfied by $\mathcal{I}$, due to our initial assumption that C_i is the smallest clause such that $\mathcal{I} \not\models C_i$. Since $x_D \notin (C_i')^{\mathcal{I}}$, from this it follows that $x_D \in (\neg D_{\forall R.D_1})^{\mathcal{I}}$ and $x_D \notin (D_{\forall R.D_1})^{\mathcal{I}}$. We obtain that $x_D \notin (D_{\forall R.D_1})^{\mathcal{I}}$ in all cases.

Since $D_{\forall R.D_1} \sqcup \forall R.D_1 \in \mathcal{N}^*$, there is also the clause $D_1 \sqcup \forall \mathsf{Inv}(R).D_{\forall R.D_1} \in \mathcal{N}^*$, either as conclusion of the role inversion rule, or because $D_{\forall R.D_1} \sqcup \forall R.D_1$ is a conclusion of the role inversion rule. From what we observed in Case 1, this clause $D_1 \sqcup \forall \mathsf{Inv}(R).D_{\forall R.D_1}$, together with the assumption that $(x_{D_2}, x_D) \in \mathsf{Inv}(R)^{\mathcal{I}}$ is due to $\exists \mathsf{Inv}(R).D \in I^{D_2}$, implies that either $x_{D_2} \in D_1^{\mathcal{I}}$ or $x_D \in (D_{\forall R.D})^{\mathcal{I}}$. We have $x_D \notin (D_{\forall R.D_1})^{\mathcal{I}}$. Therefore, $x_{D_2} \in D_1^{\mathcal{I}}$. But this contradicts that there is an edge $(x_D, x_{D_2}) \in R^{\mathcal{I}}$ with $x_{D_2} \notin D_1^{\mathcal{I}}$. Hence, $x_D \in (\forall R.D_1)^{\mathcal{I}}$ and $x_D \in C_i$.

We showed that $x_{D_2} \in D_1^{\mathcal{I}}$ in both cases. Therefore, $x \in C$ for all $x \in \Delta$ and all $C \in \mathcal{N}^*$. Hence, $\mathcal{I}$ is a model of $\mathcal{N}^*$. □

We obtain that for all sets $\mathcal{N}$ of $\mathcal{ALCHI}$ clauses with $\bot \notin \mathit{Res}^s_{\mathcal{ALCHI}}(\mathcal{N})$, we can build a model. Therefore, $\mathit{Res}^s_{\mathcal{ALCHI}}$ is refutationally complete.

Theorem 5.3.4. *$\mathit{Res}^s_{\mathcal{ALCHI}}$ provides a sound and refutationally complete decision procedure for $\mathcal{ALCHI}$ ontology satisfiability.*

Interpolation completeness of $\mathit{Int}_{\mathcal{ALCHI}}$ is established by the following theorem.

Theorem 5.3.5. *$\mathit{Int}_{\mathcal{ALCHI}}$ is interpolation complete for $\mathcal{ALCHI}\nu$.*

Proof. Let $\mathcal{O}$ be any $\mathcal{ALCHI}$ ontology, $\mathcal{S}$ any signature, $\mathcal{N} = \mathit{Cl}(\mathcal{O})$ and let $\mathcal{N}^{\mathcal{S}} = (\mathit{Int}_{\mathcal{ALCHI}}(\mathcal{N})_n)_{\mathcal{S}}$. Let C be a concept such that $\mathit{sig}(C) \subseteq \mathcal{S}$ and $\mathcal{M} = \mathit{Cl}(\exists r^*.C)$, where $r^* \notin \mathit{sig}(\mathcal{O})$. We have to show that whenever we can infer $\bot$ from $\mathcal{N} \cup \mathcal{M}$ using $\mathit{Res}^s_{\mathcal{ALCHI}}$, we can derive $\bot$ from $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ using $\mathit{Res}^s_{\mathcal{ALCHI}}$ as well (see proofs for Theorem 4.4.11 and Theorem 5.1.4).

In order to prove interpolation completeness of $\mathit{Int}_{\mathcal{ALC}}$ in Theorem 4.4.11, the following properties of $\mathit{Res}_{\mathcal{ALC}}$ and $\mathit{Int}_{\mathcal{ALC}}$ were exploited: (1) refutational completeness is preserved if only clauses with at most one negative definer literal are inferred (Theorem 4.3.3), (2) refutational completeness is preserved if we use ordered resolution

(Theorem 4.3.5), and (3) inferences with additional sets of clauses that introduce new definers can be simulated by first saturating the initial clause set with $Int_{\mathcal{ALC}}$, and then applying inferences with the second clause set (Lemma 4.4.6).

Property 2 is preserved by $Res_{\mathcal{ALCHI}}$, since we did not make any new assumptions on the ordering that underlies the model construction. For Property 3 the situation is the same as for $Int_{\mathcal{ALC}}$ and $Int_{\mathcal{ALCH}}$, since $Res^s_{\mathcal{ALCHI}}$ does not have any new role propagation rules. The only new rule in $Res_{\mathcal{ALCHI}}$ and $Int_{\mathcal{ALCHI}}$ is the role inversion rule, but this rule does not introduce new definers.

Regarding Property 1, we observe that the additional normal form transformation step necessary for $Res^s_{\mathcal{ALCHI}}$ and $Int_{\mathcal{ALCHI}}$ may introduce clauses with more than one negative definer. By computing the clause set $\mathcal{N}^{\mathcal{S}} = (Int_{\mathcal{ALCHI}}(\mathcal{N})_n)_{\mathcal{S}}$, all clauses with more than one negative definer literal are removed, since our technique for eliminating definers only works on normal clause sets (see Section 4.1.2).

The additional transformation step is performed directly on the input clause set. This means, every clause with more than one negative definer that is introduced when processing $\mathcal{N}$ due to this step is also introduced when processing $\mathcal{N}^{\mathcal{S}}$, unless the role of the corresponding universal restriction is not in $\mathcal{S}$. If the role is not in $\mathcal{S}$, the definers in question cannot interact with any definer in $\mathcal{M}$, since $\mathcal{M}$ only contains roles that are in $\mathcal{S}$. Note that inferences that use the role hierarchy are preserved by the monotonicity rules. Therefore, all clauses with more than one negative definer literal required to infer the empty clause from $\mathcal{N} \cup \mathcal{M}$ can also be generated from $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$. Hence, we can derive the empty clause from $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ if we can do so from $\mathcal{N} \cup \mathcal{M}$. This means that $\mathcal{N}^{\mathcal{S}}$ preserves all entailments in $\mathcal{S}$ of $\mathcal{N}$, and that $Ont(\mathcal{N}^{\mathcal{S}})$ is the $\mathcal{ALCHI}$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$. □

5.4 Functional Role Restrictions

The description logic $\mathcal{ALCHF}$ extends $\mathcal{ALCH}$ with functional role restrictions, which are concept expressions of the form $\leq 1r.\top$. Since $\mathcal{ALCHF}$ is closed under negation, this also makes it possible to express concepts of the form $\geq 2r.\top$, which are equivalent to negations of functional role restrictions. $\mathcal{ALCHF}$ is the first example of a language $\mathcal{L}$ where $\mathcal{L}$ uniform interpolants of $\mathcal{ALC}$ ontologies preserve more entailments than $\mathcal{ALC}$

uniform interpolants. As an example, take the following $\mathcal{ALC}$ ontology $\mathcal{O}_1$:

$$A \sqsubseteq \exists r.B$$
$$A \sqsubseteq \exists r.\neg B$$

The $\mathcal{ALC}$ uniform interpolant of $\mathcal{O}_1$ for $\{A, r\}$ is $A \sqsubseteq \exists r.\top$. This axiom entails all $\mathcal{ALC}$ concept inclusions in the signature $\{A, r\}$ that can be inferred from $\mathcal{O}_1$. However, from the fact that no individual can satisfy both B and $\neg B$ at the same time, we can deduce that every instance of A has at least two r-successors. This information cannot be expressed in $\mathcal{ALC}$, $\mathcal{ALCH}$, $\mathcal{SH}$ or $\mathcal{ALCHI}$ without using additional symbols, but it can be expressed as an $\mathcal{ALCHF}$ concept inclusion, namely as $A \sqsubseteq \geq 2r.\top$. In fact, this axiom is an $\mathcal{ALCHF}$ uniform interpolant of $\mathcal{O}_1$ for $\{A, r\}$.

The interaction between role hierarchies and functional roles allows for more problematic cases when role symbols are to be eliminated. Take as example the following ontology.

$$r_1 \sqsubseteq s$$
$$r_2 \sqsubseteq s$$
$$\top \sqsubseteq \leq 1 s.\top$$

In the signature $\mathcal{S} = \{r_1, r_2\}$, we have an infinite set of entailments of the following form, where C is any $\mathcal{ALCHF}$ concept in $\mathcal{S}$:

$$\exists r_1.C \sqcap \exists r_2.\top \sqsubseteq \exists r_2.C.$$

It is not clear how these entailments can be captured by a finite ontology, unless we allow for additional expressivity such as Boolean role constructors, in particular role conjunctions, which we are not considering in the context of this thesis. On the other hand, if the desired signature contains all role symbols that are present in the original ontology, a uniform interpolant can always be presented in $\mathcal{ALCHF}\nu$.

We first extend the normal form to account for the new expressivity.

Definition 5.4.1. An $\mathcal{ALCHF}$ *literal* is a concept of one of the following forms:

$$A \mid \neg A \mid \exists r.D \mid \forall r.D \mid \geq 2r.\top \mid \leq 1 r.\top,$$

$\exists\exists$-Role Propagation

$$\frac{C_1 \sqcup \exists R_1.D_1 \qquad C_2 \sqcup \exists R_2.D_2 \qquad R_1 \sqsubseteq_{\mathcal{N}} S \qquad R_2 \sqsubseteq_{\mathcal{N}} S}{C_1 \sqcup C_2 \sqcup \exists R_1.D_{12} \sqcup \geq 2S.\top}$$

where D_{12} is a possibly new definer representing $D_1 \sqcap D_2$.

$\geq$2-Elimination I

$$\frac{C_1 \sqcup \geq 2R.\top \qquad C_2 \sqcup \leq 1S.\top \qquad R \sqsubseteq_{\mathcal{N}} S}{C_1 \sqcup C_2}$$

$\geq$2-Elimination II

$$\frac{C_1 \sqcup \geq 2R.\top \qquad C_2 \sqcup \forall S.D \qquad \neg D \qquad R \sqsubseteq_{\mathcal{N}} S}{C_1 \sqcup C_2}$$

$\leq$1-Monotonicity

$$\frac{C \sqcup \leq 1R.\top \qquad S \sqsubseteq R}{C \sqcup \leq 1S.\top}$$

$\geq$2-Monotonicity

$$\frac{C \sqcup \geq 2R.\top \qquad R \sqsubseteq S}{C \sqcup \geq 2S.\top}$$

Figure 5.6: Additional inference rules in $Res_{\mathcal{ALCHF}}$ and $Int_{\mathcal{ALCHF}}$.

where $A \in N_c$, $r \in N_r$ and $D \in N_d$. An $\mathcal{ALCHF}$ *clause* is a role inclusion or an axiom of the form $\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n$, where $L_1, \ldots, L_n$ are $\mathcal{ALCHF}$ literals. An ontology $\mathcal{O}$ is in $\mathcal{ALCHF}$ *normal form* if every axiom in $\mathcal{O}$ is an $\mathcal{ALCHF}$ clause. Given an $\mathcal{ALCHF}$ ontology $\mathcal{O}$, we denote the $\mathcal{ALCHF}$ normal form representation of $\mathcal{O}$ by $Cl(\mathcal{O})$.

$Res^{s}_{\mathcal{ALCHF}}$ and $Int_{\mathcal{ALCHF}}$ extend respectively $Res^{s}_{\mathcal{ALCH}}$ and $Int_{\mathcal{ALCH}}$ by the rules shown in Figure 5.6, where $Res^{s}_{\mathcal{ALCHF}}$ additionally uses the $\forall\forall$-role propagation rule. As for the $\forall\exists$- and the $\forall\forall$-role propagation rules, most of these rules are motivated by the valid concept inclusion $(C_1 \sqcup C_2) \sqcap (C_3 \sqcup C_4) \sqsubseteq (C_1 \sqcup C_3 \sqcup (C_2 \sqcap C_4))$ (see Section 4.2.1).

For the $\exists\exists$-role propagation rule, assume we have a model with a domain element x that has at least one R_1-successor x_1 satisfying D_1 and at least one R_2-successor x_2 that satisfies D_2. Assume further that $R_1 \sqsubseteq_{\mathcal{N}} S$ and $R_2 \sqsubseteq_{\mathcal{N}} S$, as stated in the premise of the rule. Either $x_1 = x_2$, in which case x satisfies $\exists R_1.(D_1 \sqcap D_2)$, or $x_1 \neq x_2$, in which case x satisfies $\geq 2S.\top$. Therefore, we have $\models (C_1 \sqcup \exists R_1.D_1) \sqcap (C_2 \sqcup \exists R_2.D_2) \sqsubseteq$

$(C_1 \sqcup C_2 \sqcup \exists R_1.(D_1 \sqcap D_2) \sqcup \geq 2S.\top)$.

For the $\geq$2-elimination rule I, observe that $\geq 2R.\top \sqcap \leq 1S.\top$ is unsatisfiable in any clause set $\mathcal{N}$ with $R \sqsubseteq_{\mathcal{N}} S$. Similarly, if $\neg D \in \mathcal{N}^*$ and $R \sqsubseteq_{\mathcal{N}} S$, the concept $\geq 2R.\top \sqcap \forall S.D$ is unsatisfiable in $\mathcal{N}$, which justifies the $\geq$2-elimination rule II. In order for the latter to work properly, we have to make sure that all clauses of the form $C \sqcup \forall r.D$ with $\neg D \in \mathcal{N}^*$ are derived. It is for this reason that $Res_{\mathcal{ALCHF}}$ uses additionally the $\forall\forall$-role propagation rule from $Int_{\mathcal{ALCH}}$. The intuition of the monotonicity rules is the same as in $Res_{\mathcal{ALCH}}$.

We have the following lemma.

Lemma 5.4.2. *The calculi $Res_{\mathcal{ALCHF}}$ and $Int_{\mathcal{ALCHF}}$ are sound and terminating.*

Example 5.4.3. Assume we have the following ontology $\mathcal{O}_{\mathcal{ALCHF}}$:

$$A \sqsubseteq \exists r_1.B$$
$$A \sqsubseteq \exists r_2.\neg B$$
$$r_1 \sqsubseteq r$$
$$r_2 \sqsubseteq r$$

We want to compute the uniform interpolant for $\mathcal{S}_{\mathcal{ALCHF}} = \{A, r_1, r_2, r\}$.

We obtain the following set of clauses $\mathcal{N}_{\mathcal{ALCHF}}$:

1. $\neg A \sqcup \exists r_1.D_1$

2. $\neg D_1 \sqcup B$

3. $\neg A \sqcup \exists r_2.D_2$

4. $\neg D_2 \sqcup \neg B$

5. $r_1 \sqsubseteq r$

6. $r_2 \sqsubseteq r$

We have to infer all inferences on B, for which we have to introduce a definer representing $D_1 \sqcap D_2$. Such a definer is introduced when we apply the $\exists\exists$-role propagation rule on Clause 1 and 3.

~~7.~~ $\neg A \sqcup \geq 2r.\top \sqcup \exists r_1.D_{12}$ *($\exists\exists$-role propagation on 1, 3, 5, 6)*

~~8.~~ $\neg D_{12} \sqcup D_1$ *($D_{12} \sqsubseteq D_1$)*

~~9.~~ $\neg D_{12} \sqcup D_2$ *($D_{12} \sqsubseteq D_2$)*

Note that with the same premises, we can also infer a clause similar to Clause 7 that uses the role r_2 instead of r_1 in the last literal. In this particular case, this inference is not important, as the clause becomes redundant in a few more inference steps.

~~10.~~ $\neg D_{12} \sqcup B$ *(Resolution 2, 8)*

~~11.~~ $\neg D_{12} \sqcup \neg B$ *(Resolution 4, 9)*

12. $\neg D_{12}$ *(Resolution 10, 11)*

13. $\neg A \sqcup \geq 2r.\top$ *(∃-elimination 7, 12)*

All clauses required by the uniform interpolant have been inferred. After all definers have been eliminated, we obtain the following ontology:

$$A \sqsubseteq \exists r_1.\top$$

$$A \sqsubseteq \exists r_2.\top$$

$$A \sqsubseteq \geq 2r.\top$$

$$r_1 \sqsubseteq r$$

$$r_2 \sqsubseteq r$$

This is the $\mathcal{ALCHF}$ uniform interpolant of $\mathcal{O}_{\mathcal{ALCHF}}$ for $\mathcal{S}_{\mathcal{ALCHF}}$.

To prove the refutational completeness of $\mathcal{ALCHF}$, we have to extend several elements used in the model construction, in order to accommodate the new types of literals in the language. This time, we cannot reuse the construction of the model fragments used for $Res_{\mathcal{ALC}}$. Instead, we have to define entailment of model fragments in $\mathcal{ALCHF}$ and use a more refined ordering.

Let $\mathcal{N}^* = Res^s_{\mathcal{ALCHF}}(\mathcal{N})$ be the saturation of any set $\mathcal{N}$ of normal $\mathcal{ALCHF}$ clauses such that $\bot \notin \mathcal{N}^*$. We specify $\prec_d$ to be a total ordering on definer symbols as in Section 4.2.3, that is, $\prec_d$ is any ordering that satisfies $D_1 \prec_d D_2$ if $\neg D_1 \sqcup D_2 \in \mathcal{N}^*$. We further define a non-reflexive ordering $\prec_r$ on the role symbols such that $r \prec_r s$ iff $r \sqsubseteq_{\mathcal{N}} s$ and not $s \sqsubseteq_{\mathcal{N}} r$. $\prec_c$ is an arbitrary total ordering on the concept symbols in N_c. We define $\prec_l$ as an ordering on literals that satisfies the following constraints:

1. $A \prec_l \neg A$ for all $A \in N_c$.

2. If $A_1 \prec_c A_2$, then $\neg A_1 \prec_l A_2$

3. $\exists r.D_1 \prec_l \geq 2r.\top \prec_l \forall r.D_2 \prec_l \leq 1r.\top$ for all $r \in N_r$ and $D_1, D_2 \in N_d$.

4. $\neg D \prec_l A$ and $\neg D \prec_l \exists r.D'$, for all $D, D' \in N_d$, $A \in N_c \setminus N_d$ and $r \in N_r$.

5. If $r \prec_r s$, then $\geq 2r.\top \prec_l \geq 2s.\top$, $\leq 1r.\top \prec_l \leq 1s.\top$, $\exists r.D_1 \prec_l \exists s.D_2$ and $\forall r.D_1 \prec_l \forall s.D_2$ for all $D_1, D_2 \in N_d$.

6. If $D_1 \prec_d D_2$, then $\neg D_1 \prec_l D_2$, $\exists r.D_1 \prec_l \exists r.D_2$ and $\forall r.D_1 \prec_l \exists r.D_2$ for all $r \in N_r$.

These constraints fully correspond to the constraints we put on the ordering used in the original model construction (see page 65), only that instead of one ordering $\prec_s$ on symbols in $(N_c \cup N_r) \setminus N_d$, we now use two separate orderings for concept and for role symbols. Whereas for the description logics we considered until now, the ordering between role symbols is arbitrary, the current ordering depends on the role hierarchy. For the same reasons as for the original ordering, an ordering fulfilling these constraints therefore always exists.

In order to incorporate the expressivity of $\mathcal{ALCHF}$, the definition of model fragments and satisfiability in model fragments has to be adapted as well.

Definition 5.4.4. A *model fragment literal* is a concept of the forms A, $\exists r.D$, $\geq 2r.\top$ and $\geq 2r.D$, where $A \in N_c$, $r \in N_r$ and $D \in N_d$. A *model fragment* is a set of model fragment literals. Given a model fragment I and an $\mathcal{ALCHF}$ literal L, I *satisfies* L, in symbols $I \models L$, if one of the following conditions hold:

1. $L = A$, $A \in N_c$, $A \in I$.

2. $L = \neg A$, $A \in N_c$, and $A \notin I$.

3. $L = \exists r.D$, $r \in N_r, D \in N_d$, and $\exists s.D' \in I$ or $\geq 2s.D' \in I$, where $s \sqsubseteq_{\mathcal{N}} r$ and either $D' = D$ or $\neg D' \sqcup D \in \mathcal{N}^*$.

4. $L = \forall r.D$, $r \in N_r, D \in N_d$, and there are no model fragment literals of the forms $\exists s.D'$, $\geq 2s.D'$ or $\geq 2r.\top$ in I such that $s \sqsubseteq_{\mathcal{N}} r$, $D' \neq D$ and $\neg D' \sqcup D \notin \mathcal{N}^*$.

5. $L = \geq 2r.\top$ and (1) there are two model fragment literals of the form $\exists s.D$ in I such that $s \sqsubseteq_{\mathcal{N}} r$, or (2) there is a model fragment literal of the form $\geq 2s.\top$ or $\geq 2s.D$ in I such that $s \sqsubseteq_{\mathcal{N}} r$.

6. $L = \leq 1r.\top$, there are no model fragment literals of the forms $\geq 2s.D$ or $\geq 2s.\top$ in I such that $s \sqsubseteq_{\mathcal{N}} r$, and all model fragment literals of the form $\exists s.D' \in I$ with $s \sqsubseteq_{\mathcal{N}} r$ have the same definer D'.

Given a model fragment I and a clause C, we say I *satisfies* C, in symbols $I \models C$, if there is a literal $L \in C$ such that $I \models L$.

Except for the fact that this definition incorporates literals of the form $\geq 2r.\top$ and $\leq 1r.\top$ there are two notable changes in this definition. First, we use a notion of model fragment literals which differs from $\mathcal{ALCHF}$ literals in that it also covers concepts of the form $\geq 2r.D$. In former sections, we only used literals in the model fragments that are concepts in the language under consideration. Second, the definition of satisfiability of clauses takes into account the role hierarchy in $\mathcal{N}$. For the model constructions used in earlier sections, it is not necessary to take the role hierarchy into account, since literals of the form $\exists s.D$ are added for every literal $\exists r.D$ with $s \sqsubseteq_{\mathcal{N}} r$ if the clause set is saturated using the $\exists$-monotonicity rule. This ensures that all role axioms of the clause set are satisfied by candidate model. However, when constructing a candidate model for clause sets with functional role restrictions, we have to ensure that only as few literals of the forms $\exists r.D$ and $\geq 2r.\top$ as needed are added.

We describe how model fragments I^D for definers $D \in N_d$ and for the special symbol ϵ are constructed. As before, if $\neg D \in \mathcal{N}^*$, $I^D = \emptyset$. If $\neg D \notin \mathcal{N}^*$, I^D is defined incrementally as follows.

1. $I^D_0 = \emptyset$ if $D = \epsilon$, $I^D_0 = \{D\}$ if $D \in N_d$.

2. If $I^D_{i-1} \models C_i$, then $I^D_i = I^D_{i-1}$, otherwise:

   (a) If the maximal literal in C_i is of the form A, then $I^D_i = I^D_{i-1} \cup \{A\}$.

   (b) If the maximal literal in C_i is of the form $\exists R.D'$, then $I^D_i = I^D_{i-1} \cup \{\exists R.D'\}$.

   (c) If the maximal literal in C_i is of the form $\geq 2R.\top$, and $\exists R'.D' \in I^D_{i-1}$ for some $R' \sqsubseteq_{\mathcal{N}} R$ and $D' \in N_d$, then $I^D_i = (I^D_{i-1} \setminus \{\exists R.D'\}) \cup \{\geq 2R.D'\}$.

   (d) If the maximal literal in C_i is of the form $\geq 2R.\top$, and there is no literal $\exists R'.D' \in I^D$ for any $R' \sqsubseteq_{\mathcal{N}} R$ and $D' \in N_d$, then $I^D_i = I^D_{i-1} \cup \{\geq 2R.\top\}$.

   (e) If the maximal literal in C_i is of the form $\forall R.D'$, then I^D_i is the result of replacing every literal of the form $\geq 2R'.\top$ in I^D_j, where $R' \sqsubseteq_{\mathcal{N}} R$, by $\geq 2R'.D'$.

3. $I^D = I^D_n$, where $n = |\mathcal{N}^*|$.

As for the model construction presented in Section 4.2.3, we show the construction monotone.

Lemma 5.4.5. *If $I^D_i \models C_i$ then $I^D_j \models C_i$ for all $j > i$.*

Proof. Assume $I^D_i \models C_i$. We show that in this case $I^D_j \models C_i$ for all $j > i$. If $I^D_i \models C_i$, there must be a literal $L \in C_i$ such that $I^D_i \models L$. We distinguish the different cases for L.

1. Assume L is of the form A. Then $A \in I^D_i$, and no step in the construction removes literals of the form A. Therefore, $I^D_j \models A$ and also $I^D_j \models C_i$ for all $j > i$.

2. Assume L is of the form $\neg A$. Then $\neg A \in I^D_i$. A is only added to a subsequent model fragment I^D_j if there is a clause C_j with $C_j \prec_c C_i$ in which the maximal literal is A. Such a clause cannot exist with the constraints of the ordering, and therefore $I^D_j \models L$ for all $j > i$.

3. Assume L is of the form $\exists R.D_1$. Then, I^D_i contains a literal of the form $\exists S.D_2$ or $\geq 2S.D_2$, where $S \sqsubseteq_{\mathcal{N}} R$ and either $D_2 = D_1$ or $\neg D_1 \sqcup D_2 \in \mathcal{N}^*$. If $\geq 2S.D_2 \in I^D_i$, then $\geq 2S.D_2 \in I^D_j$ for all $j > i$, since literals of this form are never removed from the current model fragment. If $\exists S.D_2 \in I^D_i$, for all $j > i$ we either have $\exists S.D_2 \in I^D_j$ or $\geq 2S.D_2 \in I^D_j$ (due to Step 3c in the construction of model fragments). In both cases, $I^D_j \models L$ and $I^D_j \models C$.

4. Assume L is of the form $\geq 2R.\top$. Then either $\geq 2S.\top \in I^D_i$ or $\geq 2S.D \in I^D_i$, where $S \sqsubseteq_{\mathcal{N}} R$. In the latter case, $\geq 2S.D \in I^D_j$ for all $j > i$. In the former case, it is possible that $\geq 2S.\top$ is replaced by a literal of the form $\geq 2S.D$ (see Step 3e in the construction of model fragments), in which case $\geq 2S.\top$ is still satisfied. We obtain $I^D_j \models L$ and $I^D_j \models C$ for all $j > i$.

5. Assume L is of the form $\forall R.D_1$ and let $j > i$. We show $I^D_j \models \forall R.D_1$ by contradiction and assume $I^D_j \not\models \forall R.D_1$. If $I^D_j \not\models \forall R.D_1$, I^D_j must contain a literal L_2 of the form $\exists S.D_2$, $\geq 2S.\top$ or $\geq 2S.D_2$, where $S \sqsubseteq_{\mathcal{N}} R$, $D_2 \neq D_1$ and $\neg D_2 \sqcup D_1 \notin \mathcal{N}^*$.

(a) Assume L_2 is of the form $\exists S.D_2$ or $\geq 2S.\top$, and $L_2 \in I_j^D$. Then, there must be a clause C_k with $i < k \leq j$ such that L_2 is the maximal literal in C_k. However, since $S \sqsubseteq_{\mathcal{N}} R$, we have $L_2 \prec_l L$. Therefore, such a clause cannot exist.

(b) Assume L_2 is of the form $\geq 2S.D_2$. Such a literal can only be in I_j^D if there is a clause C_k, $i < k \leq j$, such that the maximal literal in C_k is of the form $\forall S.D_2$, and $\geq 2S.\top \in I_k^D$ (see Step 3e of the construction of model fragments). As observed earlier, $\geq 2S.\top$ is only added for clauses smaller than C_i. But then, since $I_i^D \models L$ by assumption, $\geq 2S.\top \notin I_i^D$. We obtain that $\geq 2S.\top \notin I_k^D$ for any $k > i$, and consequently $\geq 2S.\top$ cannot be replaced by another literal for C_k. Therefore, $I_j^D \models \forall R.D_1$ and $I_j^D \models C$.

6. Assume L is of the form $\leq 1R.\top$ and let $j > i$. $I_j^D \not\models L$ if there are two literals of the form $\exists R_1.D_1$, $\exists R_2.D_2 \in I_j^D$, where $R_1 \sqsubseteq_{\mathcal{N}} R$ and $R_2 \sqsubseteq_{\mathcal{N}} R$, or if there is one literal of the form $\geq 2S.\top \in I_j^D$ or $\geq 2S.D' \in I_j^D$. If there is a literal L' of the forms $\exists S.D'$ or $\geq 2S.\top$ in $(I_j^D \setminus I_i^D)$, where $S \sqsubseteq_{\mathcal{N}} R$, there must be a clause C_k with $k > i$ in which L' is maximal. Since $S \sqsubseteq_{\mathcal{N}} R$, we have $L' \prec_l L$, and therefore, such a clause cannot exist. For the same reason, we cannot have $\geq 2S.D \in I_j^D$. We obtain $I_j^D \models L$ and $I_j^D \models C$ for all $j > i$, which contradicts our initial assumption.

We obtain for every clause C_i, $I_i^D \models C_i$ implies $I_j^D \models C_i$ for all $j > i$. □

Lemma 5.4.6. *If $I_i^D \not\models C_i$, then for all $j > i$ and $C_i' \subseteq C_i$, $I_j^D \not\models C_i'$.*

Proof. Since for all $C_i' \subseteq C_i$, $I_j^D \not\models C_i$ implies $I_j^D \not\models C_i'$, it is sufficient to consider the case where $C_i' = C_i$.

Assume there is a clause C_i with $I_i^D \not\models C_i$. We do the proof by contradiction and assume there is a model fragment I_j^D with $j > i$ such that $I_j^D \models C_i$. If $I_j^D \models C_i$, there must be a literal $L \in C_i$ such that $I_j^D \models L$. We distinguish the different cases for L.

1. If L is of the form A, $\exists R.D'$ or $\geq 2R.\top$, there must be a clause C_k, $k > i$, where A, $\exists S.D'$ or $\geq 2S.\top$ is maximal and $S \sqsubseteq_{\mathcal{N}} R$. With the constraints of the ordering, this is impossible.

2. Assume L is of the form $\neg A$. If $I_i^D \not\models \neg A$, then $A \in I_i^D$. The model construction does not remove literals of the form A from subsequent model fragments, and therefore $I_j^D \not\models \neg A$ for all $j > i$, which contradicts the initial assumption.

3. Assume L is of the form $\leq 1R.\top$. If $I_i^D \not\models L$, either there are two literals of the form $\exists S_1.D_1, \exists S_2.D_2 \in I_i^D$ such that $S_1 \sqsubseteq_{\mathcal{N}} R$, $S_2 \sqsubseteq_{\mathcal{N}} R$ and $D_1 \neq D_2$, there is a literal $\geq 2S.D' \in I_i^D$ with $S \sqsubseteq_{\mathcal{N}} R$, or there is a literal $\geq 2S.\top \in I_i^D$. Only a literal of the latter form might be removed for a subsequent clause, but only to be replaced by a literal $\geq 2S.D'$, in which case $\leq 1R.\top$ is still not satisfied by the model fragment. We obtain $I_j^D \not\models L$, which contradicts our initial assumption.

4. Assume L is of the form $\forall R.D_1$. Then, I_i^D contains a literal L_1 of the form $\exists S.D_2$, $\geq 2S.\top$ or $\geq 2S.D_2$, where $S \sqsubseteq_{\mathcal{N}} R$, $D_1 \neq D_2$ and $\neg D_2 \sqcup D_1 \notin \mathcal{N}^*$. L_1 cannot be of the form $\exists S.D_2$ or $\geq 2S.D_2$, since then also $L_1 \in I_j^D$. Therefore, L_1 is of the form $\geq 2S.\top$. $\geq 2S.\top \notin I_j^D$ then implies that there is a clause C_k, $i < k \leq j$, in which the maximal literal is of the form $\forall S'.D_2$, $S \sqsubseteq_{\mathcal{N}} S'$, $\neg D_2 \sqcup D_1 \in \mathcal{N}^*$, and for which $I_{k-1}^D \not\models C_k$, since then $\geq 2S.\top$ is replaced in I_k^D by $\geq 2S.D_2$. Note that $\neg D_2 \sqcup D_1$ implies that D_2 was introduced by an application of a role propagation rule. Since D_1 occurs in the literal $\forall R.D_1$ and D_2 occurs in the literal $\forall S'.D_2$, this must have been an application of the $\forall\forall$-role propagation rule. We obtain that that $S' \sqsubseteq_{\mathcal{N}} R$. But then $\forall S'.D_2 \prec_l \forall R.D_2$, and $\forall S'.D_2$ cannot be maximal in a clause larger than C_i, which contradicts earlier observations.

We obtain for every clause C_i that $I_i^D \not\models C_i$ implies $I_j^D \not\models C_i$ for all $j > i$. □

Using these lemmata, we can show that for any definer D or ϵ, given that $\neg D \notin \mathcal{N}^*$, the constructed model fragments I^D satisfy all clauses in $\mathcal{N}^*$.

Lemma 5.4.7. *Let D any definer with $\neg D \notin \mathcal{N}^*$ or $D = \epsilon$. Then, $I^D \models C$ for all $C \in \mathcal{N}^*$.*

Proof. We do the proof by contradiction and assume C_i is the smallest clause for which $I^D \not\models C_i$. Denote the maximal literal in C_i by L and let $C_i = C_i' \sqcap L$. Because $I^D \not\models C_i$, also $I^D \not\models L$. We distinguish the cases for L.

1. L is of the form A, $\exists R.D'$ or $\geq 2R.\top$. Then $I^D \models L$ follows directly from the model construction and Lemma 5.4.5.

2. L is of the form $\neg A$. Since the model construction is monotone, we have a contradiction for both $A \neq D$ and $A = D$ for the same reasons as in the corresponding proof for Lemma 4.2.15.

3. L is of the form $\forall R_1.D_1$. $I^D \not\models L$ implies that I^D contains a literal of the form $\exists R_2.D_2$, $\geq 2R_2.\top$ or $\geq 2R_2.D_2$ with $R_2 \sqsubseteq_{\mathcal{N}} R_1$, $D_1 \neq D_2$ and $\neg D_2 \sqcup D_1 \notin \mathcal{N}^*$. We distinguish the cases for these different forms of literals.

   (a) If $\exists R_2.D_2 \in I^D$, there must be a clause $C_j = C'_j \sqcup \exists R_2.D_2$, in which $\exists R_2.D_2$ is maximal and for which $I^D_{j-1} \not\models C_j$. Together with Lemma 5.4.6, $I^D_i \not\models C'_j$. Due to the $\forall\exists$-role propagation rule, there are also the clauses $C_k = C'_i \sqcup C'_j \sqcup \exists R_2.D_{12}, \neg D_{12} \sqcup D_1$ and $\neg D_{12} \sqcup D_2$ in $\mathcal{N}^*$. $C_j \prec_c C_i$, since for the maximal literals we have $\exists R_2.D_2 \prec_l \forall R_1.D_1$. Because $\exists R_2.D_{12} \prec_c \exists R_2.D_2$, also $C_k \prec C_j$. However, because $I^D_{j-1} \not\models C_j$, we cannot have $\exists R_2.D_{12} \in I^D$. With $I^D \not\models C'_i$ and $I^D \not\models C'_j$, we obtain $I^D \not\models C_k$, which contradicts the initial assumption that C_i is the smallest clause for which $I^D \not\models C_i$.

   (b) Assume $\geq 2R_2.\top \in I^D$. Then, there must be a clause $C_j = C'_j \sqcup \geq 2R_2.\top$, where $\geq 2R_2.\top$ is maximal and $I^D_{j-1} \not\models C_j$. Observe that $\geq 2R_2.\top \prec_l \forall R_1.D_1$. Therefore, if $\geq 2R_2.\top \in I^D_j$, also $\geq 2R_2.\top \in I^D_{i-1}$. But then, due to Step 3e in the model fragment construction, in I^D_i, $\geq 2R_2.\top$ is replaced by $\geq 2R_2.D_1$. Hence, $\geq 2R_2.\top \in I^D$ is impossible.

   (c) Assume there is a literal $\geq 2R_2.D_2 \in I^D$ with $R_2 \sqsubseteq_{\mathcal{N}} R_1$, $D_2 \neq D_1$ and $\neg D_2 \sqcup D_1 \notin \mathcal{N}^*$. Observe that since $I^D \not\models C_i$, by Lemma 5.4.5 also $I^D_i \not\models C_i$. Therefore, $\geq 2R_2.D_2$ must already be contained in I^D_i, and there must be a clause $C_j = C'_j \sqcup \forall R_3.D_2 \in \mathcal{N}^*$ smaller than C_i with $I^D_{j-1} \not\models C_j$ and $R_2 \sqsubseteq_{\mathcal{N}} R_3$, in which $\forall R_3.D_2$ is maximal. Because of Lemma 5.4.6, this implies $I^D \not\models C'_j$. Due to the $\forall\forall$-role propagation rule and since $R_2 \sqsubseteq_{\mathcal{N}} R_1$ and $R_2 \sqsubseteq_{\mathcal{N}} R_3$, we then also have a clause $C_k = C'_i \sqcup C'_j \sqcup \forall R_2.D_{12} \in \mathcal{N}^*$, with $\neg D_{12} \sqcup D_1$, $\neg D_{12} \sqcup D_2 \in \mathcal{N}^*$. $C_k \prec_c C_j$, and since $I^D \not\models C'_i$, $I^D \not\models C'_j$ and $\geq 2R_2.D_2 \in I^D$, we further obtain that $I^D \not\models C_k$. This contradicts our initial assumption that C_i is the smallest clause with $I^D \not\models C_i$.

4. $L = \leq 1R_1.\top$. There are two possibilities.

(a) $\geq 2R_2.\top \in I^D$ or $\geq 2R_2.D' \in I^D$, where $R_2 \sqsubseteq_{\mathcal{N}} R_1$. In each case, there then must be a clause $C_j = C'_j \sqcup \geq 2R_2.\top \in \mathcal{N}^*$ with $I^D_{j-1} \not\models C_j$. This implies $I^D \not\models C'_j$ due to Lemma 5.4.6. The $\geq$2-elimination rule I produces the clause $C_k = C'_i \sqcup C'_j$ from C_i and C_j, and $C_k \prec_c C_i$. Since $I^D \not\models C_i$ and $I^D \not\models C'_j$, also $I^D \not\models C_k$, which contradicts the assumption that C_i is the smallest clause with $I^D \not\models C_i$.

(b) There are two literals $\exists R_2.D_2, \exists R_3.D_3 \in I^D$ with $D_2 \neq D_3$, $R_2 \sqsubseteq_{\mathcal{N}} R_1$ and $R_3 \sqsubseteq_{\mathcal{N}} R_1$. This implies the existence of two clauses $C_j = C'_j \sqcup \exists R_2.D_2$, $C_k = C'_k \sqcup \exists R_3.D_3$ with $I^D_{j-1} \not\models C_j$ and $I^D_{k-1} \not\models C_k$. We then also have $I^D \not\models C'_j$ and $I^D \not\models C'_k$, due to Lemma 5.4.6, and $\neg D_2 \sqcup D_3 \notin \mathcal{N}^*$ and $\neg D_3 \sqcup D_2 \notin \mathcal{N}^*$, since otherwise either $I^D_{j-1} \models C_j$ or $I^D_{k-1} \models C_k$. $\exists\exists$-role propagation on C_j and C_k produces the clause $C'_j \sqcup C'_k \sqcup \exists R_2.D_{23} \sqcup \geq 2R_1.\top$ with $\neg D_{23} \sqcup D_2, \neg D_{23} \sqcup D_3 \in \mathcal{N}^*$, and via $\geq$2-elimination I with C_i the clause $C_l = C'_i \sqcup C'_j \sqcup C'_k \sqcup \exists R_2.D_{23}$. $\exists R_2.D_{23}$ cannot be maximal in C_l, since otherwise $C_l \prec_c C_j$, $I^D_l \models \exists R_2.D_{23}$, and $\exists R_2.D_2$ is not added to I^D. But then $I^D \not\models \exists R_2.D_{23}$, and because also $I^D \not\models C'_i \sqcup C'_j \sqcup C'_k$, we have $I^D \not\models C_l$. Furthermore, $C_l \prec_c C_i$, because $\leq 1R_1.\top \notin C_l$ and all other literals in C_l are smaller than $\leq 1R_1.\top$. But then the initial assumption that C_i is the smallest clause with $I^D \not\models C_i$ is contradicted.

There can be no smallest clause C_i such that $I^D \not\models C_i$, and hence, we obtain $I^D \models C_i$ for all clauses $C_i \in \mathcal{N}^*$. □

In order to satisfy the new types of literals of the form $\geq 2r.D$ that can occur in model fragments, we also have to adapt the construction of the final candidate model, which we do by using two domain elements per definer instead of one. The candidate model $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ is defined as follows.

1. $\Delta^{\mathcal{I}} = \{x^1_D, x^2_D \mid D \in \mathcal{N}_d \cup \{\epsilon\}\}$
2. For all $A \in N_c$, $A^{\mathcal{I}} = \{x^1_D, x^2_D \mid A \in I^D\}$
3. For all $r \in N_r$, $r^{\mathcal{I}} = \{(x^i_{D_1}, x^1_{D_2}) \mid i \in \{1,2\}, \exists r'.D_2 \in I^{D_1}, r' \sqsubseteq_{\mathcal{N}} r\}$
   $\cup \{(x^i_{D_1}, x^j_{D_2}) \mid i,j \in \{1,2\}, \geq 2r'.D_2 \in I^{D_1}, r' \sqsubseteq_{\mathcal{N}} r\}$
   $\cup \{(x^i_D, x^j_\epsilon) \mid i,j \in \{1,2\}, \geq 2r'.\top \in I^D, r' \sqsubseteq_{\mathcal{N}} r\}$.

The model construction is similar to the one for $\mathcal{ALC}$, only that two individuals x_D^1 and x_D^2 are used for each definer D in order to satisfy literals of the form $\geq 2r.D$, and two individuals x_ϵ^1 and x_ϵ^2 are used to satisfy literals of the form $\geq 2r.\top$.

In addition, since the set of clauses is not saturated using the $\exists$-monotonicity rule this time, we explicitly make sure that role inclusion axioms $r \sqsubseteq s$ are satisfied. It is therefore easy to verify that $\mathcal{I}$ is a model for $\mathcal{N}^*$. This allows us to establish refutational completeness of $Res^s_{\mathcal{ALCHF}}$.

Theorem 5.4.8. *$Res^s_{\mathcal{ALCHF}}$ is terminating, sound and refutationally complete.*

For interpolation completeness, observe that Lemma 4.4.6, which is concerned with combined definers, also holds for $Int_{\mathcal{ALCHF}}$. Since $Int_{\mathcal{ALCHF}}$ has a rule for any possible combination of role restrictions, whether two definers D_1 and D_2 are combined solely depends on the roles under which they occur, and the negative definer literals that are in the same clause as the role restrictions. Note that the additional set $\mathcal{M}$ used to prove interpolation completeness in Theorem 5.4.10 and Theorem 5.1.4 cannot contain role inclusions, since $\mathcal{M}$ is the clausal normal form representation of an axiom of the form $\top \sqsubseteq \exists r^*.C$.

However, another property we exploited for proving interpolation completeness of $Int_{\mathcal{ALC}}$ and $Int_{\mathcal{ALCH}}$ is that we can choose an arbitrary ordering $\prec_s$ between role and concept symbols and refine the corresponding calculi $Res_{\mathcal{ALC}}$ and $Res_{\mathcal{ALCH}}$ based on this ordering. For $Res^s_{\mathcal{ALCHF}}$, this is not possible, since such an ordering has to take into consideration the role hierarchy. However, it is still possible to refine $Res^s_{\mathcal{ALCHF}}$ in such a way that inferences on concept symbols $A \notin \mathcal{S}$ are preferred over inferences on other literals. For this reason, $Int_{\mathcal{ALCHF}}$ is interpolation complete for forgetting concept symbols, that is, it can be used to compute uniform interpolants for every signature $\mathcal{S}$ with $N_r \subseteq \mathcal{S}$.

Theorem 5.4.9. *$Int_{\mathcal{ALCHF}}$ is interpolation complete for forgetting concept symbols in $\mathcal{ALCHF}\nu$.*

Note that if an ontology does not contain role inclusion axioms, an ordering can be chosen independent of the role symbols. Therefore, we can establish that $Int_{\mathcal{ALCHF}}$ is interpolation complete for $\mathcal{ALCF}\nu$.

Theorem 5.4.10. *$Int_{\mathcal{ALCHF}}$ is interpolation complete for $\mathcal{ALCF}\nu$.*

Example 5.4.11 (Role Forgetting in $\mathcal{ALCF}$)**.** Consider the following ontology $\mathcal{O}_{\mathcal{ALCF}}$.

$$A_1 \sqsubseteq \exists r.B_1$$

$$A_2 \sqsubseteq \exists r.B_2$$

$$B_1 \sqcap B_2 \sqsubseteq \bot$$

$$\top \sqsubseteq \leq 1 r.\top$$

We want to compute the $\mathcal{ALCF}$ uniform interpolant for $\mathcal{S}_{\mathcal{ALCF}} = \{A_1, A_2\}$. We obtain the following normal form $\mathcal{N}_{\mathcal{ALCF}} = Cl(\mathcal{O}_{\mathcal{ALCF}})$:

1. $\neg A_1 \sqcup \exists r.D_1$
2. $\neg D_1 \sqcup B_1$
3. $\neg A_2 \sqcup \exists r.D_2$
4. $\neg D_2 \sqcup B_2$
5. $\neg B_1 \sqcup \neg B_2$
6. $\leq 1 r.\top$

We begin by applying the $\exists\exists$-role propagation rule on Clause 1 and Clause 2, which leads to the introduction of a new definer D_{12}.

7. $\neg A_1 \sqcup \neg A_2 \sqcup \exists r.D_{12} \sqcup \geq 2 r.\top$ *($\exists\exists$-Role Propagation 1, 3)*
8. $\neg D_{12} \sqcup D_1$ *($D_{12} \sqsubseteq D_1$)*
9. $\neg D_{12} \sqcup D_2$ *($D_{12} \sqsubseteq D_2$)*
10. $\neg D_{12} \sqcup B_1$ *(Resolution 2, 8)*
11. $\neg D_{12} \sqcup B_2$ *(Resolution 4, 9)*
12. $\neg D_{12} \sqcup \neg B_2$ *(Resolution 5, 10)*
13. $\neg D_{12}$ *(Resolution 11, 12)*

We also have to infer all entailments in the signature from clauses involving the role symbol r. This is achieved using the $\exists$-elimination rule and the functional role restriction resolution rule.

14. $\neg A_1 \sqcup \neg A_2 \sqcup \geq 2 r.\top$ *($\exists$-Elimination 7, 13)*
15. $\neg A_1 \sqcup \neg A_2$ *(Functional Role Restriction Resolution 6, 14)*

After eliminating all definer symbols, we obtain the following ontology, which is the $\mathcal{ALCF}$ uniform interpolant of $\mathcal{O}_{\mathcal{ALCF}}$ for $\mathcal{S}_{\mathcal{ALCF}}$.

$$A_1 \sqcap A_2 \sqsubseteq \bot$$

5.5 Bringing It All Together: $\mathcal{SIF}$

As mentioned in the introduction, in order to obtain a refutationally and interpolation complete calculus for $\mathcal{SIF}$, it is not sufficient to combine the rules from $Res^s_{\mathcal{SH}}$, $Res^s_{\mathcal{ALCHF}}$ and $Res^s_{\mathcal{ALCHI}}$. Role hierarchies bring a problem of a different nature when combined with functional role restrictions and inverse roles, which we will discuss shortly at the end of this chapter. Functional role restrictions have to be treated differently when combined with inverse roles. The following example illustrates this.

Example 5.5.1. Consider the following $\mathcal{ALCIF}$ axiom, which contains both inverse roles and functional role restrictions.

$$A \sqsubseteq \exists r.(\exists r^{-}.\neg A \sqcap \leq 1 r^{-}.\top)$$

We argue that A is unsatisfiable. The axiom states that every instance of A must have an r-successor that has an r-predecessor satisfying $\neg A$. Moreover, this r-successor has at most one r-predecessor, which can only be the instance of A itself. In other words, if A could have an instance, this instance would also have to satisfy $\neg A$, which is a contradiction.

The interplay between inverse roles is crucial for the unsatisfiability of A. However, the only rule in $Int_{\mathcal{ALCHI}}$ that deals with inverse roles applies to universal restrictions exclusively, of which there are none in this axiom. Hence, the combined rules of $Int_{\mathcal{ALCHI}}$ and $Int_{\mathcal{ALCHF}}$ are not sufficient to infer the unsatisfiability of A.

The problem is that the rules for functional role restrictions we have seen so far only deal with existential role restrictions and number restrictions, whereas the rule for inverse roles only deals with universal restrictions. This way, the rules cannot interact, and the combined calculi cannot properly detect interactions between functional and inverse roles. One way to overcome this might be to add rules dealing with inverse roles occurring in existential role restrictions and number restrictions. However, it turns out

Universalisation Rule:

$$\frac{C_1 \sqcup \exists R.D \qquad C_2 \sqcup \leq 1R.\top}{C_1 \sqcup C_2 \sqcup \forall R.D}$$

Figure 5.7: Universalisation rule in $Res_{\mathcal{SIF}}$ and $Int_{\mathcal{SIF}}$.

that an easier approach is to add an additional rule for functional role restrictions. This rule is the universalisation rule shown in Figure 5.7.

The universalisation rule derives from a functional role restriction and an existential role restriction a universal one. The conclusion of this rule can be used both in the premise of the role inversion rule and in the premise of the transitivity rule. Soundness of the rule can be argued as follows. If a domain element x in an interpretation $\mathcal{I}$ has an R-successor that satisfies D ($x \in (\exists R.D)^{\mathcal{I}}$), and if x has at most one R-successor ($x \in (\leq 1R.\top)^{\mathcal{I}}$), then every R-successor of x satisfies D, since there is only one ($x \in (\forall R.D)^{\mathcal{I}}$). Therefore, the following concept inclusion is valid, which verifies the soundness of the universalisation rule:

$$(\exists R.D \sqcap \leq 1R.\top) \sqsubseteq \forall R.D$$

We adapt the definition of the normal form, which additionally makes sure that transitivity axioms are correctly handled.

Definition 5.5.2. A $\mathcal{SIF}$ *literal* is a concept of the form A, $\neg A$, $\exists R.D$, $\forall R.D$, $\leq 1R.\top$ or $\geq 2R.\top$, where $A \in N_c$, $D \in N_d$, R is of the form r or r^- and $r \in N_r$. A $\mathcal{SIF}$ *clause* is an RBox axiom of the form $\mathsf{trans}(R)$, where R is a role, or a TBox axiom of the form $\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n$, where each L_i, $1 \leq i \leq n$, is a $\mathcal{SIF}$ literal. A $\mathcal{SIF}$ ontology $\mathcal{N}$ is in $\mathcal{SIF}$ *normal form* if every axiom in $\mathcal{N}$ is a $\mathcal{SIF}$ clause, and if for every clause $\mathsf{trans}(R) \in \mathcal{N}$, there is a corresponding clause $\mathsf{trans}(\mathsf{Inv}(R)) \in \mathcal{N}$.

In the proof for refutational completeness of $Res^s_{\mathcal{ALCHI}}$, we used of Lemma 5.3.2, which states that for every universal restriction $\forall R.D$ occurring in $\mathcal{N}^* = Res_{\mathcal{ALCHI}}(\mathcal{N})$, there is a unique clause of the form $D_{\forall R.D} \sqcup \forall R.D \in \mathcal{N}^*$ assigning the definer $D_{\forall R.D}$ to $\forall R.D$. This lemma could be proved because in $Res^s_{\mathcal{ALCHI}}$, the role inversion rule is the only rule that infers universal restriction literals that are not already present in the input clause set.

In $Res_{\mathcal{SIF}}$, we have three additional rules that introduce universal restriction literals, namely the transitivity rule, the universalisation rule and the $\forall\forall$-role propagation rule. The latter is only part of the calculus to enable inference of the $\geq$2-elimination rule II, and can be ignored in this context. The other rules on the other hand produce universal restrictions that are needed in the premise of the role inversion rule. In order to preserve a property like the one stated in Lemma 5.3.2, we therefore impose a specific ordering in which rules are applied. As a result, the input clause set is processed in several stages, where redundancy elimination can be applied in any stage:

1. The initial clause set $\mathcal{N}$ is obtained by transforming the input ontology $\mathcal{O}$ into the clause set $Cl(\mathcal{O})$, where for each clause $\mathsf{trans}(R)$, the clause $\mathsf{trans}(\mathsf{Inv}(R))$ is added.

2. The clause set $\mathcal{N}^{\forall}$ is obtained by saturating $\mathcal{N}$ using the transitivity and the universalisation rule.

3. The clause set $\mathcal{N}^{D_\forall}$ is obtained from $\mathcal{N}^{\forall}$ by adding for every clause $C \sqcup \forall R.D$ in $\mathcal{N}^{\forall}$ the clauses $\neg D_{\forall R.D} \sqcup C$ and $D_{\forall R.D} \sqcup \forall R.D$.

4. $Res_{\mathcal{SIF}}(\mathcal{N})$ is obtained from $\mathcal{N}^{D_\forall}$ by saturating $\mathcal{N}^{D_\forall}$ using all rules of $Res_{\mathcal{SIF}}$ that have not been used in the second stage, that is, all rules of $Res_{\mathcal{SIF}}$ except the transitivity and the universalisation rule. In other words, $Res_{\mathcal{SIF}}(\mathcal{N})$ is obtained from $\mathcal{N}^{D_\forall}$ using the rules of $Res_{\mathcal{ALCHI}}$ and $Res_{\mathcal{ALCHF}}$.

The following example illustrates the different stages of the calculus.

Example 5.5.3. We apply $Res_{\mathcal{SIF}}$ on the axiom given in Example 5.5.1.

$$A \sqsubseteq \exists r.(\leq 1 r^{-}.\top \sqcap \exists r^{-}.\neg A)$$

In Stage 1, we translate this axiom into normal form $\mathcal{N}_{\mathcal{ALCIF}}$:

1. $\neg A \sqcup \exists r.D_1$
2. $\neg D_1 \sqcup \leq 1 r^{-}.\top$
3. $\neg D_1 \sqcup \exists r^{-}.D_2$
4. $\neg D_2 \sqcup \neg A$

In Stage 2, we saturate $\mathcal{N}_{\mathcal{ALCIF}}$ using the transitivity rule and the universalisation rule. In this example, the only rule that can be applied is the universalisation rule. $\mathcal{N}^\forall$ has therefore the following additional clause:

5. $\neg D_1 \sqcup \forall r^-.D_2$ *(Universalisation 2, 3)*

In Stage 3, we use the second structural transformation step to add additional clauses for every universal restriction occurring in $\mathcal{N}^\forall_{\mathcal{ALCIF}}$. In our example, this is the universal restriction in Clause 5. As a result, $\mathcal{N}^{D_\forall}_{\mathcal{ALCIF}}$ contains additionally the following clauses:

6. $\neg D_3 \sqcup \neg D_1$
7. $D_3 \sqcup \forall r^-.D_2$

We are now in the last stage, Stage 4, in which we apply all remaining rules of the calculus.

8. $D_2 \sqcup \forall r.D_3$ *(Role Inversion 7)*
9. $\neg A \sqcup \forall r.D_3$ *(Resolution 4, 8)*
10. $\neg A \sqcup \exists r.D_{13}$ *(∀∃-Role Propagation 1, 9)*
11. $\neg D_{13} \sqcup D_1$ *($D_{13} \sqsubseteq D_1$)*
12. $\neg D_{13} \sqcup D_3$ *($D_{13} \sqsubseteq D_3$)*
14. $\neg D_{13} \sqcup \neg D_1$ *(Resolution 6, 14)*
15. $\neg D_{13}$ *(Resolution 11, 14)*
16. $\neg A$ *(∃-Elimination 11, 16)*

We have inferred that A is unsatisfiable, which is as expected.

5.6 Refutational Completeness of $Res^s_{\mathcal{SIF}}$

In contrast to the description logics we considered so far, $\mathcal{SIF}$ does not have the finite model property. This is illustrated by the following ontology $\mathcal{O}^\infty$:

$$\top \sqsubseteq \exists s.B$$

$$B \sqsubseteq \neg A \sqcap \exists r.A$$

$$A \sqsubseteq \exists r.A$$

$$\top \sqsubseteq \leq 1 r^-.\top$$

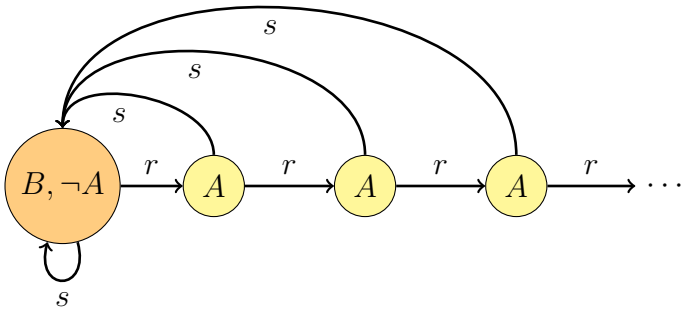

Figure 5.8: Infinite model of $\mathcal{O}^\infty$.

$\mathcal{O}^\infty$ is satisfiable, but it only has infinite models, such as the one shown in Figure 5.8.

In the model construction approach we used so far, each definer D in the clause set is represented by one or two domain elements. For ontologies like $\mathcal{O}^\infty$, this approach has to fail, since the number of definers in a saturated set of clauses is always finite. We illustrate this problem with an example.

The saturation $Res^s_{\mathcal{SIF}}(\mathcal{N}^\infty)$, where $\mathcal{N}^\infty$ is the normal form representation of $\mathcal{O}^\infty$, is the following, where elements of $\mathcal{N}^\infty$ use bold face numbers:

1. $\exists s.D_1$

2. $\neg D_1 \sqcup B$

3. $\neg B \sqcup \neg A$

4. $\neg B \sqcup \exists r.D_2$

5. $\neg D_2 \sqcup A$

6. $\neg A \sqcup \exists r.D_2$

7. $\leq 1 r^-.\top$

8. $\neg D_1 \sqcup \neg A$ *(Resolution 2, 3)*

9. $\neg D_1 \sqcup \exists r.D_2$ *(Resolution 2, 4)*

10. $\neg D_2 \sqcup \neg B$ *(Resolution 3, 5)*

11. $\neg D_2 \sqcup \exists r.D_2$ *(Resolution 5, 6)*

The model constructions presented in Section 4.2.3 and Section 5.4 both create the

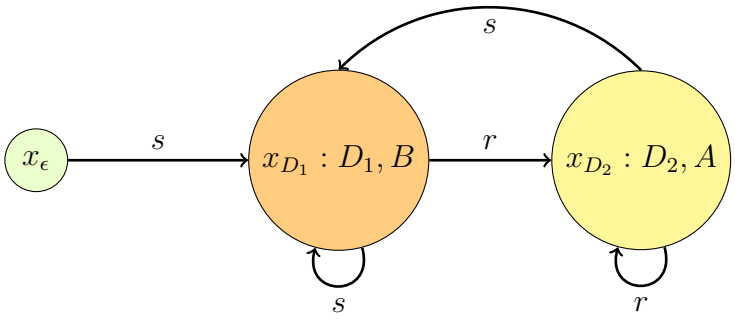

Figure 5.9: Model candidate for $\mathcal{N}^*$ generated by method for $Res_{\mathcal{ALC}}$.

following model fragments for these clauses:

$$I^\epsilon = \{\exists s.D_1\}$$
$$I^{D_1} = \{D_1, B, \exists r.D_2, \exists s.D_1\}$$
$$I^{D_2} = \{D_2, A, \exists r.D_2, \exists s.D_1\}$$

As a result, we obtain the candidate model shown in Figure 5.9. The domain element x_{D_2} has two incoming r-edges, one from x_{D_1} and one from itself, and does therefore not satisfy $\leq 1 r^-.\top$. Hence, it is not a model of $\mathcal{O}^\infty$.

The problem with the previous model construction approaches is that we reuse domain elements. In order to satisfy existential role restrictions of the form $\exists r.D$ in the model fragments, we always use the same domain element x_D as r-successor. An alternative approach is to build the candidate model incrementally, starting from a root element x_ϵ, and add new domain elements for every existential role restriction in the corresponding model fragment, possibly expanding the candidate model into an infinite tree.

We describe this incremental model construction in detail. Model fragments are created using the technique for $Res^s_{\mathcal{ALCHF}}$ described in Section 5.4. Moreover, we define a modification function on model fragments. This is necessary due to pairs of clauses of the form $\neg D_1 \sqcup C_1 \sqcup \exists r.D_2$ and $\neg D_2 \sqcup C_2 \sqcup \exists r^-.D_1$. Suppose we have a domain element x_{D_1} for D_1, and we add an r-successor x_{D_2} based on I^{D_2} to satisfy $\neg D_1 \sqcup C_1 \sqcup \exists r.D_2$. The clause $\neg D_2 \sqcup C_2 \sqcup \exists r^-.D_1$ is already satisfied by x_{D_2}, since x_{D_2} has x_{D_1} as an r-predecessor satisfying D_1. Accordingly, we do not need to add another r^--successor for x_{D_2}.

It is impossible to determine which existential role restrictions can be ignored if each

model fragment is processed independently. In our example, since we build the candidate model incrementally, it might be that the domain element x_{D_2} for D_2 is created before any domain element for D_1, in which case we do have to add an r^--successor. It is however crucial that we do not add more successors to any element than necessary, since this can lead to concepts of the form $\leq 1 r.\top$ not being satisfied by the candidate model.

To represent this mechanism concisely, we define a function $\dot{\leftarrow}$ that takes as arguments two model fragments I_1, I_2 and a role R, and returns a new model fragment $I_1 \dot{\leftarrow}^R I_2$. Intuitively, $I_1 \dot{\leftarrow}^R I_2$ represents individuals with an incoming R-edge from an individual that satisfies the literals in I_2. $I_1 \dot{\leftarrow}^R I_2$ is defined as follows:

$$I_1 \dot{\leftarrow}^R I_2 = I_1 \setminus \{\exists \mathsf{Inv}(R).D \mid D \in I_2, \exists R.D' \in I_2 \text{ or } \geq 2 R.D' \in I_2, D' \in I_1\}$$

If there is a domain element x_2 satisfying the literals in I_2, and an R-edge between x_2 and another domain element x_1, then x_1 already satisfies $\exists \mathsf{Inv}(R).D$, given $D \in I_2$. Therefore, we can ignore the literal $\exists \mathsf{Inv}(R).D$ in the model construction.

Furthermore, we define the union $\mathcal{I}_1 \cup \mathcal{I}_2$ of two interpretations $\mathcal{I}_1$ and $\mathcal{I}_2$. Given the interpretations $\mathcal{I}_1 = \langle \Delta^{\mathcal{I}_1}, \cdot^{\mathcal{I}_1} \rangle$ and $\mathcal{I}_2 = \langle \Delta^{\mathcal{I}_2}, \cdot^{\mathcal{I}_2} \rangle$, we define the union $\mathcal{I}_1 \cup \mathcal{I}_2$ of $\mathcal{I}_1$ and $\mathcal{I}_2$ as $\langle \Delta^{\mathcal{I}_1 \cup \mathcal{I}_2}, \cdot^{\mathcal{I}_1 \cup \mathcal{I}_2} \rangle$, where $\Delta^{\mathcal{I}_1 \cup \mathcal{I}_2}$ and $\cdot^{\mathcal{I}_1 \cup \mathcal{I}_2}$ are defined as follows for all $A \in N_c$ and $r \in N_r$:

$$\Delta^{\mathcal{I}_1 \cup \mathcal{I}_2} = \Delta^{\mathcal{I}_1} \cup \Delta^{\mathcal{I}_2}$$
$$A^{\mathcal{I}_1 \cup \mathcal{I}_2} = A^{\mathcal{I}_1} \cup A^{\mathcal{I}_2}$$
$$r^{\mathcal{I}_1 \cup \mathcal{I}_2} = r^{\mathcal{I}_1} \cup r^{\mathcal{I}_2}$$

We now describe the construction of the candidate model $\mathcal{I}$. The candidate model is defined using the function $\mathcal{I}(x, I)$, which takes a domain element x and a model fragment and returns a tree-shaped interpretation. x is the root of this tree, and I contains the literals x should satisfy in the interpretation. The definition of $\mathcal{I}(x, I)$ is inductive in the sense that it depends on other interpretations $\mathcal{I}(x_D, I_D)$ for each definer D which occurs in a literal of the form $\exists R.D$ or $\geq 2 R.D$.

Given a domain element x and a model fragment I, the interpretation $\mathcal{I}(x, I)$ is

defined as follows:

$$\Delta^{\mathcal{I}(x,I)} = \Delta^{\mathcal{I}^{succ(I,x)}} \cup \{x\},$$

$$A^{\mathcal{I}(x,I)} = \begin{cases} A^{\mathcal{I}^{succ(I,x)}} & \text{if } A \notin I \\ A^{\mathcal{I}^{succ(I,x)}} \cup \{x\} & \text{if } A \in I. \end{cases}$$

$$r^{\mathcal{I}(x,I)} = r^{\mathcal{I}^{succ(I,x)}}$$
$$\cup \{(x, x_D) \mid \exists r.D \in I\} \cup \{(x_D, x) \mid \exists r^-.D \in I\}$$
$$\cup \{(x, x^1_D), (x, x^2_D) \mid \geq 2r.D \in I, \} \cup \{(x^1_D, x), (x^2_D, x) \mid \exists r^-.D \in I\},$$

where the domain elements x_D, x^1_D and x^2_D refer to the respective domain elements in the *successor interpretation union* $\mathcal{I}^{succ(I,x)}$, which is defined as follows:

$$\mathcal{I}^{succ(x,I)} = \bigcup_{\exists R.D \in I} \mathcal{I}(x_D, I^D \xleftarrow{R} I) \cup \bigcup_{\geq 2R.D \in I} \Big(\mathcal{I}(x^1_D, I^D \xleftarrow{R} I) \cup \mathcal{I}(x^2_D, I^D \xleftarrow{R} I)\Big),$$

where it is assumed that the different interpretations involved in the successor interpretation union have disjoint domains. Intuitively, the successor interpretation union $\mathcal{I}^{succ(x,I)}$ is the union of *successor interpretations* $\mathcal{I}(x_D, I^D \xleftarrow{R} I)$, where D is referenced in I via a role restriction on R.

Given $\mathcal{I} = \mathcal{I}(x, I)$, we refer to x as the *root element* of $\mathcal{I}$, and I as the *initial model fragment* of $\mathcal{I}$. The definition of $\mathcal{I}(x, I)$ is inductive, since it refers via the successor interpretation union to the successor interpretations $\mathcal{I}(x_D, I^D \xleftarrow{R} I)$ for any definer for which there is a literal $\exists R.D$ in the initial model fragment. The domain $\Delta^{\mathcal{I}(x,I)}$ therefore consists of all domain elements present in any successor interpretation, plus the domain element x.

Due to the inductive nature of this definition, $x_D \in \mathcal{I}(x_D, I^D \xleftarrow{R} I)$, and therefore there is one domain element $x_D \in \mathcal{I}(x, I)$ for every existential restriction $\exists R.D \in I$. The interpretation for each concept symbol A contains the successor interpretations of A and, if $A \in I$, additionally the root element x. This way it is ensured that x satisfies all concept symbols in the model fragment. Note that for every satisfiable definer D, $D \in I^D$. Therefore, the definition ensures that $x_D \in D^{\mathcal{I}(x_D, I^D \xleftarrow{R} I)}$ and $x_D \in D^{\mathcal{I}(x,I)}$. Recall that due to Lemma 4.2.16, no model fragment contains a literal $\exists R.D$ for a definer D with $\neg D \in \mathcal{N}^*$. For this reason, the model construction never uses domain elements x_D if the corresponding definer D is unsatisfiable.

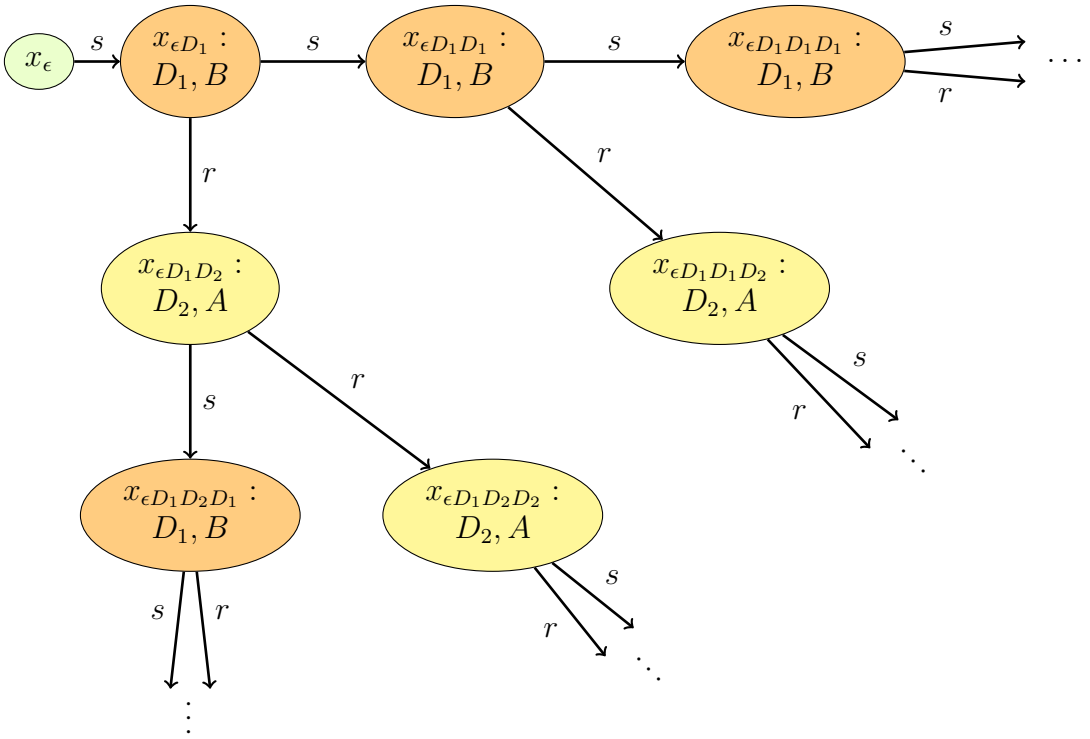

Figure 5.10: Model candidate for $\mathcal{N}^{\infty}*$ built using the new method.

Similar to the interpretation of concept symbols, for each role symbol r the interpretation takes the successor interpretations of r, and adds a relation (x, x_D) for every $\exists r.D$. Since $x_D \in D$, this ensures that $x \in (\exists r.D)^{\mathcal{I}(x,I)}$ if $\exists r.D \in I$. The same holds for inverse roles and literals of the form $\geq 2R.D$.

The special definer ϵ serves as the root of the candidate model. This candidate model is defined as $\mathcal{I} = \mathcal{I}(x_\epsilon, I^\epsilon)$. For the clause set $\mathcal{N}^{\infty *}$ given earlier, the corresponding interpretation is shown in Figure 5.10. Even though this interpretation is more complex than the model shown in Figure 5.8, every element has at most one r-predecessor. Therefore, every element satisfies $\leq 1 r^-.\top$. Indeed, the interpretation is a model of $\mathcal{N}^{\infty *}$. The graph additionally visualises an important property of interpretations generated this way. They have the form of a tree with the root element as its root, and every domain element x_D has at most one edge that is not due to the elements in I^D.

We prove that also in the general case, $\mathcal{I}(x_\epsilon, I^\epsilon)$ is a model for all clauses in $\mathcal{N}^*$ that are not transitivity axioms.

Lemma 5.6.1. *Let $\mathcal{N}$ be any set of $\mathcal{SIF}$ clauses, $\mathcal{N}^* = Res^s_{\mathcal{SIF}}(\mathcal{N})$, and $\mathcal{I} = \mathcal{I}(x_\epsilon, I^\epsilon)$*

the candidate model built using the above construction based on the clauses in $\mathcal{N}^*$. Then, $\mathcal{I} \models C$ for every clause $C \in \mathcal{N}^*$ that is not of the form $\mathsf{trans}(R)$.

Proof. The proof by contradiction. Let C be the smallest clause, according to $\prec_c$, for which $\mathcal{I} \not\models C$. Then, there is a domain element $x \in \Delta^{\mathcal{I}}$ such that $x \notin C^{\mathcal{I}}$. Let $\mathcal{I}_x = \mathcal{I}(x, I_x)$ be the corresponding interpretation that has x as a root element.

We do a case analysis based on the maximal literal L in C, where $C = C' \sqcup L$. Observe that since $x \notin C^{\mathcal{I}}$, we also have $x \notin L^{\mathcal{I}}$.

1. L is of the form A or $\neg A$. A contradiction arises for the same reasons as in the refutational completeness proof of $Res_{\mathcal{ALC}}$ (see Lemma 4.2.15).

2. L is of the form $\exists R.D$ or $\geq 2r.\top$. We cannot have $\exists R.D \in I_x$, since then the model construction would ensure that $x \in (\exists R.D)^{\mathcal{I}}$, which contradicts our assumption. Therefore, $\exists R.D \notin I_x$. The construction of model fragments I^D ensures that every clause is satisfied by I^D (Lemma 5.4.7). Therefore, I_x must be the result of the operator $\dot{\leftarrow}$, that is, $I_x = I^{D'} \xleftarrow{\mathsf{Inv}(R)} I_p$, $D \in I_p$, and $\exists R.D$ has been removed from $I^{D'}$ for I_x. This implies that $\mathcal{I}_x$ is a successor interpretation of an interpretation $\mathcal{I}_p = \mathcal{I}(x_p, I_p)$, and that there is a literal $\exists \mathsf{Inv}(R).D' \in I_p$ such that $D' \in I_x$. In this case, there is an edge $(x, x_p) \in R^{\mathcal{I}}$. If $D \in I_p$, I_x does not contain the literal $\exists R.D$. But then also $x_p \in D^{\mathcal{I}}$ and $x \in (\exists R.D)$. This contradicts $x \notin L^{\mathcal{I}}$.

3. L is of the form $\forall R.D$. $x \notin (\forall R.D)^{\mathcal{I}}$ implies that there is an edge $(x, x_2) \in R^{\mathcal{I}}$ with $x_2 \notin D^{\mathcal{I}}$. This edge can be created in two ways by the model construction. (a) There is a literal $\exists R.D' \in I_x$ with $D' \neq D$ and $\neg D' \sqcup D \notin \mathcal{N}^*$. In this case, the situation is the same as for $Res_{\mathcal{ALCH}}$, and we have a contradiction for the same reason as in the proof for Lemma 5.4.7. (b) There is a predecessor interpretation $\mathcal{I}_p = \mathcal{I}(I_p, x_p)$ of $\mathcal{I}_x$ with $\exists \mathsf{Inv}(R).D' \in I_p$ and $D' \in I_x$. Then, the situation is the same as for $Res^s_{\mathcal{ALCHI}}$ (see Theorem 5.3.4), since the only new rules that introduce universal restrictions $\forall R.D$ that do not already occur in $\mathcal{N}$ are the transitivity rule and the universalisation rule, and these are applied in the first stage, before the additional introduction of definer symbols. Therefore, both cases contradict the assumption that $\mathcal{I} \not\models C$.

4. L is of the form $\leq 1R.\top$. $x \notin (\leq 1R.\top)^{\mathcal{I}}$ implies that there are two different individuals x_1 and x_2 with $(x, x_1), (x, x_2) \in R^{\mathcal{I}}$.

   (a) x_1 and x_2 are both root nodes of a successor interpretation of $\mathcal{I}_x$. This implies that I_x contains two literals $\exists R.D_1$ and $\exists R.D_2$, or one literal of the form $\geq 2R.D$. We then have a contradiction for the same reasons as in the proof for Lemma 5.4.7, which was used to show refutational completeness of $Res^s_{\mathcal{ALCHF}}$.

   (b) x_1 is the root node of an interpretation $\mathcal{I}_1 = \mathcal{I}(I_1, x_1)$, of which $\mathcal{I}_x$ is a successor interpretation. Note that due to the tree-shape of the interpretations built, $\mathcal{I}_x$ cannot be the successor interpretation of a second interpretation. Therefore, x_2 must be the root node of a successor interpretation $\mathcal{I}_2 = \mathcal{I}(I_2, x_2)$ of $\mathcal{I}_x$. Consequently, $\exists \mathsf{Inv}(R).D_1 \in I_1$ or $\geq 2\mathsf{Inv}(R).D_1 \in I_1$, where $D_1 \in I_x$, and $\exists R.D_2 \in I_x$, where $D_2 \in I_2$. Note that $\geq 2R_2.D_2 \in I_x$ is impossible, since we already showed that this leads to a contradiction.

   Since $\exists R.D_2 \in I_x$, there is a clause $C'_2 \sqcup \exists R.D_2 \in \mathcal{N}^*$ such that $\exists R.D_2$ is maximal and $I_x \not\models C_2$, due to the monotonicity of the model construction of model fragments for $Res^s_{\mathcal{ALCHF}}$ (see Lemma 5.4.6).

   Observe that $\leq 1R.\top$ must occur as literal already in the original clause set $\mathcal{N}$, possibly in a different clause $C'' \sqcup \leq 1R.\top$, since there is no rule that infers new functional role restrictions. Similarly, the literal $\exists R.D_2$ either occurs in a clause in $\mathcal{N}$, or can be generated by subsequent $\forall\exists$-role propagation rules on a clause $C''_2 \sqcup \exists R.D'_2$ present in $\mathcal{N}$. (If $\exists R.D_2$ occurs in the conclusion of the existential role combination rule, since $C' \sqcup \leq 1R.\top \in \mathcal{N}^*$, the universalisation rule applies to one of the premises, deriving a clause where the existential role restriction is replaced by a universal role restriction, so that an alternative derivation infers $\exists R_2.D_2$ using only universal role propagations.) Either $D'_2 = D_2$ or $\neg D_2 \sqcup D'_2 \in \mathcal{N}^*$.

   $C'' \sqcup \leq 1R.\top$ and $C''_2 \sqcup \exists R'_2.D'_2$ are both in the initial clause set $\mathcal{N}$. Hence, in Stage 2, the clause $C'' \sqcup C''_2 \sqcup \forall R.D'_2$ is derived. Because of the second definer introduction in Stage 3, we have the clauses $D_3 \sqcup \forall R.D'_2$ and $\neg D_3 \sqcup C'' \sqcup C''_2 \in \mathcal{N}^*$. A similar sequence of rule applications that infers

$C' \sqcup \leq 1R.\top$ from $C'' \sqcup \leq 1R.\top$ can be used to infer $\neg D_3 \sqcup C' \sqcup C''_2$ from $\neg D_3 \sqcup C'' \sqcup C''_2$. Analogously, one can derive the clause $\neg D_3 \sqcup C' \sqcup C'''_2$ such that $C'''_2 \subseteq C'_2$. (The subset relation holds since the $\forall\exists$-role propagation rule may add additional literals that are not in C'_2.) Since $I_x \not\models C'$ and $I_x \not\models C_2$, and since $\neg D_3 \sqcup C' \sqcup C''_2 \prec_c C$, this establishes $I_x \not\models D_3$.

Because $\exists \mathsf{Inv}(R).D_1 \in I_1$, there is a clause $C_1 \sqcup \exists \mathsf{Inv}(R).D_1$ such that $I_1 \not\models C_1$. Due to the role inversion rule and $D_3 \sqcup \forall R.D'_2$, $\mathcal{N}^*$ contains the clause $D'_2 \sqcup \forall \mathsf{Inv}(R).D_3$. The $\forall\exists$-role propagation rule infers $C_1 \sqcup D'_2 \sqcup \exists \mathsf{Inv}(R).D_{13}$, $\neg D_{13} \sqcup D_1$, $\neg D_{13} \sqcup D_3 \in \mathcal{N}^*$. As noted earlier, either $D'_2 = D_2$ or $\neg D_2 \sqcup D'_2 \in \mathcal{N}^*$. In both cases, $C_1 \sqcup D_2 \sqcup \exists \mathsf{Inv}(R).D_{13} \in \mathcal{N}^*$: in the first case because $D'_2 = D_2$, in the second case by resolution on D'_2.

Since $I_x \not\models D_3$, and $I_x \models \neg D_{13} \sqcup D_3$ (the clause is smaller than C), we have $I_x \not\models D_{13}$. The only way in which $x_1 \in (\exists \mathsf{Inv}(R).D_{13})^{\mathcal{I}}$ can be true is therefore due to another successor x' with $(x_1, x') \in \mathsf{Inv}(R)^{\mathcal{I}}$ with $x' \in D^{\mathcal{I}}_{13}$. But this is successor is only generated if $I_1 \models \exists \mathsf{Inv}(R).D_{13}$, in which case $\exists \mathsf{Inv}(R).D_1$ is not added to I_1 in the model fragment construction. We therefore establish $I_1 \not\models \exists \mathsf{Inv}(R).D_{13}$. We have $C_1 \sqcup D_2 \sqcup \exists \mathsf{Inv}(R).D_{13} \in \mathcal{N}^*$. $C_1 \sqcup D_2 \sqcup \exists \mathsf{Inv}(R).D_{13} \prec_c C$, which due to our original assumption implies $I_1 \models C_1 \sqcup D_2 \sqcup \exists \mathsf{Inv}(R).D_{13}$. This, together with $I_1 \not\models C_1$ and $I_1 \not\models \exists \mathsf{Inv}(R).D_{13}$, allows us to conclude that $I_1 \models D_2$.

$I_1 \models D_2$, and $\mathcal{I}_x$ is a successor interpretation of $\mathcal{I}_1$ via the role $\mathsf{Inv}(R)$. Therefore, $I_x = I_x \xleftarrow{\mathsf{Inv}(R)} I_1$. By definition, this means I_x contains no element of the set $\{\exists R.D' \mid D' \in I_x, \exists \mathsf{Inv}(R).D'' \in I_1 \text{ or } \geq 2\mathsf{Inv}(R).D'' \in I_1, D'' \in I_1\}$. Set $D' = D_2$ and $D'' = D_1$, and we obtain that $\exists R_2.D_2 \notin I_x$. But this contradicts that x has a second R-successor x_2, which means $\leq 1R.\top$ is satisfied by x.

We established that $x \notin (\leq 1R.D)^{\mathcal{I}}$ leads to a contradiction in all cases.

All cases for maximal literals in C lead to a contradiction, which means there cannot be a smallest clause $C \in \mathcal{N}^*$ such that $\mathcal{I} \not\models C$. Hence, we have for all clauses $C \in \mathcal{N}^*$ that are not of the form $\mathsf{trans}(R)$ that $\mathcal{I} \models C$. □

In order to extend $\mathcal{I}$ to an interpretation $\mathcal{I}^{\mathsf{trans}}$ that also satisfies the transitivity

axioms in $\mathcal{N}^*$, we use the same trick as in Section 5.2, which is by adding the transitive closure to transitive roles. Note that transitive roles that are transitive are not allowed to occur in concepts of the form $\leq 1R.\top$ (see Section 3.1). The additional edges in $\mathcal{I}^{\mathsf{trans}}$ therefore do not affect the satisfaction of concepts of the form $\leq 1R.\top$. Hence, $\mathcal{I}^{\mathsf{trans}}$ satisfies all clauses in $\mathcal{N}^*$, and we have the following lemma.

Lemma 5.6.2. *$\mathcal{I}^{\mathsf{trans}}$ satisfies all clauses in $\mathcal{N}^*$.*

Based on this lemma, we can establish refutational completeness of $Res^s_{\mathcal{SIF}}$. Interpolation completeness of $Int_{\mathcal{SIF}}$ can be established in the same way as for the earlier calculi, since the only new rule, the universalisation rule, does not introduce new definers nor clauses with more than one negative definer. Note also that due to the absence of role hierarchies in $Int_{\mathcal{SIF}}$, forgetting role symbols does not create a problem.

Theorem 5.6.3. *$Res^s_{\mathcal{SIF}}$ is sound and refutationally complete, and provides a decision procedure for $\mathcal{SIF}$ ontology satisfiability.*

Theorem 5.6.4. *$Int_{\mathcal{SIF}}$ is interpolation complete.*

5.6.1 On the Incompleteness for $\mathcal{SHIF}$

It might be surprising that we did not include role hierarchies in the calculus presented in this section, as all the other calculi apply to description logics with role hierarchies. It turns out that role hierarchies cause an additional problem when combined with both inverse role and functional role restrictions, which is not easily solved with a calculus of the kind we are using throughout the thesis. This is illustrated by the following ontology $\mathcal{O}_{\mathcal{SHIF}}$:

$$
\begin{array}{ll}
A \sqsubseteq \exists r_1.B & r_1 \sqsubseteq s_{13} \\
B \sqsubseteq \exists r_2^-.C & r_3 \sqsubseteq s_{13} \\
C \sqsubseteq \exists r_3.D & r_2 \sqsubseteq s_{23} \\
A \sqsubseteq \exists r_4.E & r_3 \sqsubseteq s_{23} \\
\top \sqsubseteq \leq 1 s_{13}^-.\top & r_2 \sqsubseteq s_{24} \\
\top \sqsubseteq \leq 1 s_{23}.\top & r_4 \sqsubseteq s_{24} \\
\top \sqsubseteq \leq 1 s_{24}.\top & D \sqcap E \sqsubseteq \bot
\end{array}
$$

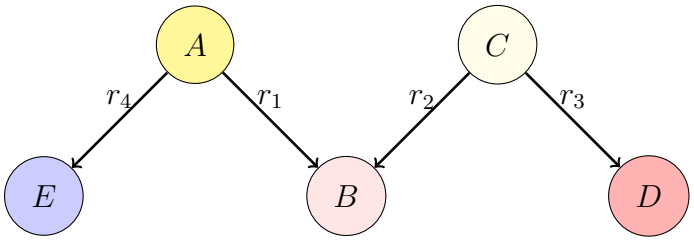

Figure 5.11: Interpretation of $\mathcal{O}_{\mathcal{SHIF}}$ that does not satisfy the role hierarchy.

Without the role hierarchy, the interpretation in Figure 5.11 would be model for $\mathcal{O}_{\mathcal{SHIF}}$. Together with the role hierarchy and the functional role restrictions, we see that in a model, the nodes for B and C would have to collapse into one node, and so would the nodes for E, B and D. Therefore, every instance of A would have a successor that satisfies both D and E. However, due to the last axiom in the ontology, such a successor cannot exist in a model for $\mathcal{O}_{\mathcal{SHIF}}$. From this it follows that A is unsatisfiable in $\mathcal{O}_{\mathcal{SHIF}}$. Our calculus, even if extended with monotonicity rules, is not able to detect this unsatisfiability, and it cannot be used to prove that $\top \sqsubseteq \neg A$ is an entailment. The problem is that while our calculus is able to transfer information about definer literals using the role propagation roles, it cannot propagate information about role symbols in the same way. This problem does not only affect our calculus, but also shows the incompleteness of other saturation based reasoning methods for description logics that have been published in the literature. The method presented in Kazakov (2009), defined for the description logic Horn $\mathcal{SHIQ}$, which is expressive enough to represent this ontology, also fails to detect the unsatisfiability of A. A solution might be to use a language that can represent conjunctions of roles, which we did not consider in the context of this thesis.

We conclude the chapter with an overview of the rules introduced in this chapter, which is shown in Figure 5.12 and Figure 5.13. For each rule, we mention for which calculus it was first introduced, and whether its application is restricted to Stage 2 of the derivation.

Resolution *($Res_{\mathcal{ALC}}$)*

$$\frac{C_1 \sqcup A \qquad C_2 \sqcup \neg A}{C_1 \sqcup C_2}$$

$\exists$-Elimination *($Res_{\mathcal{ALC}}$)*

$$\frac{C_1 \sqcup \exists R.D \qquad \neg D}{C_1}$$

$\exists$-Monotonicity *($Res_{\mathcal{ALCH}}$)*

$$\frac{C \sqcup \exists R.D \qquad R \sqsubseteq S}{C \sqcup \exists S.D}$$

$\forall$-Monotonicity *($Int_{\mathcal{ALCH}}$)*

$$\frac{C \sqcup \forall R.D \qquad S \sqsubseteq R}{C \sqcup \forall S.D}$$

Role hierarchy *($Int_{\mathcal{ALCH}}$)*

$$\frac{S \sqsubseteq R \qquad R \sqsubseteq T}{S \sqsubseteq T}$$

$\forall\exists$-Role Propagation *($Res^{s}_{\mathcal{ALCH}}$)*

$$\frac{C_1 \sqcup \forall R.D_1 \qquad C_2 \sqcup \exists S.D_2 \qquad S \sqsubseteq_{\mathcal{N}} R}{C_1 \sqcup C_2 \sqcup \exists S.D_{12}}$$

where D_{12} is a possibly new definer representing $D_1 \sqcap D_2$.

$\forall\forall$-Role Propagation *($Int_{\mathcal{ALCH}}$, $Res_{\mathcal{ALCHF}}$)*

$$\frac{C_1 \sqcup \forall R_1.D_1 \qquad C_2 \sqcup \forall R_2.D_2 \qquad S \sqsubseteq_{\mathcal{N}} R_1 \qquad S \sqsubseteq_{\mathcal{N}} R_2}{C_1 \sqcup C_2 \sqcup \forall S.D_{12}}$$

where D_{12} is a possibly new definer representing $D_1 \sqcap D_2$.

Transitivity *($Res_{\mathcal{SH}}$, only used in Stage 2 in $Res_{\mathcal{SIF}}$)*

$$\frac{C \sqcup \forall R.D \qquad \mathsf{trans}(S) \qquad S \sqsubseteq_{\mathcal{N}} R}{C \sqcup \forall S.D' \quad \neg D' \sqcup D \quad \neg D' \sqcup \forall S.D'}$$

Figure 5.12: Complete set of rules introduced in this chapter, Part 1.

Role Inversion *($Res_{\mathcal{ALCHI}}$)*

$$\frac{D_1 \sqcup \forall R.D_2}{D_2 \sqcup \forall \mathsf{Inv}(R).D_1}$$

$\exists\exists$-Role Propagation *($Res_{\mathcal{ALCHF}}$)*

$$\frac{C_1 \sqcup \exists R_1.D_1 \qquad C_2 \sqcup \exists R_2.D_2 \qquad R_1 \sqsubseteq_{\mathcal{N}} S \qquad R_2 \sqsubseteq_{\mathcal{N}} S}{C_1 \sqcup C_2 \sqcup \exists R_1.D_{12} \sqcup \geq 2S.\top}$$

where D_{12} is a possibly new definer representing $D_1 \sqcap D_2$.

$\geq$2-Elimination I *($Res_{\mathcal{ALCHF}}$)*

$$\frac{C_1 \sqcup \geq 2R.\top \qquad C_2 \sqcup \leq 1S.\top \qquad R \sqsubseteq_{\mathcal{N}} S}{C_1 \sqcup C_2}$$

$\geq$2-Elimination II *($Res_{\mathcal{ALCHF}}$)*

$$\frac{C_1 \sqcup \geq 2R.\top \qquad C_2 \sqcup \forall S.D \qquad \neg D \qquad R \sqsubseteq_{\mathcal{N}} S}{C_1 \sqcup C_2}$$

Universalisation *($Res_{\mathcal{SIF}}$, only used in Stage 2)*

$$\frac{C_1 \sqcup \exists R.D \qquad C_2 \sqcup \leq 1R.\top}{C_1 \sqcup C_2 \sqcup \forall R.D}$$

Figure 5.13: Complete set of rules introduced in this chapter, Part 2.

Chapter 6

Uniform Interpolation with Cardinality Restrictions

In Chapter 5, we extended the calculi $Res_{\mathcal{ALC}}$ and $Int_{\mathcal{ALC}}$ stepwise to develop a uniform interpolation method for $\mathcal{SIF}$. In this chapter, we generalise the rules of these calculi in order to deal with number restrictions. $Int_{\mathcal{ALCHF}}$ has several rules that deal with the interactions between literals of the form $\exists R.D$, $\forall R.D$, $\leq 1R.\top$ and $\geq 2R.\top$. In $\mathcal{SHQ}$, these literals can be equivalently represented using only two types of role restrictions, namely number restrictions of the form $\geq nr.D$ and $\leq nr.D$. These number restrictions also cover all new concepts we can express in $\mathcal{SHQ}$. Rather than extending $Int_{\mathcal{ALCHF}}$ by new rules, we therefore formulate a new calculus with fewer rules, which generalises the rules of $Int_{\mathcal{ALCHF}}$.

In Section 5.4, we observed that due to functional role restrictions, an $\mathcal{ALCHF}$ uniform interpolant can preserve more information from the input ontology than an $\mathcal{ALCH}$ uniform interpolant. This is even more the case with qualified number restrictions, as we illustrate with the following $\mathcal{ALC}$ ontology $\mathcal{O}_{bike}$.

$$\begin{aligned} \mathsf{Bicycle} &\sqsubseteq \exists \mathsf{hasWheel}.\mathsf{FrontWheel} \sqcap \exists \mathsf{hasWheel}.\mathsf{RearWheel} \\ \mathsf{FrontWheel} &\sqsubseteq \mathsf{Wheel} \sqcap \neg \mathsf{RearWheel} \\ \mathsf{RearWheel} &\sqsubseteq \mathsf{Wheel} \sqcap \neg \mathsf{FrontWheel} \end{aligned}$$

Since RearWheel and FrontWheel are disjoint concepts, we can infer from this ontology that each bicycle has at least two wheels. Accordingly, an $\mathcal{SHQ}$ uniform interpolant

of $\mathcal{O}_{bike}$ for $\mathcal{S}_{bike} = \{\mathsf{Bicycle}, \mathsf{hasWheel}, \mathsf{Wheel}\}$ is the following:

$$\mathsf{Bicycle} \sqsubseteq \geq 2\mathsf{hasWheel}.\mathsf{Wheel}$$

By contrast, for $\mathcal{ALCHF}$ and $\mathcal{ALCH}$, the uniform interpolants are expressed in the following axioms, which preserve less information:

$$\mathsf{Bicycle} \sqsubseteq \geq 2\mathsf{hasWheel}.\top \sqcap \exists \mathsf{hasWheel}.\mathsf{Wheel} \qquad (\mathcal{ALCHF})$$

$$\mathsf{Bicycle} \sqsubseteq \exists \mathsf{hasWheel}.\mathsf{Wheel} \qquad (\mathcal{ALCH})$$

In applications where numbers of successors are important, $\mathcal{SHQ}$ uniform interpolation can therefore also contribute if the ontology is expressed in a less expressive description logic.

Even though $\mathcal{SHF}$ and $\mathcal{SHQ}$ are of the same complexity class for standard reasoning tasks (Tobies, 2001), number restrictions interact in more complex ways than the role restrictions we have seen so far. This becomes intuitive when one looks at the first-order logic representations of number restrictions (see Section 3.1):

$$\top \sqsubseteq \geq nr.D \iff \forall x \exists y_1, \ldots \exists y_n \Big[\Big(\bigwedge_{1 \leq i \leq n} \Big(r(x, y_i) \wedge D(y_i) \Big) \wedge \bigwedge_{1 \leq i < j \leq n} y_i \neq y_j \Big) \Big]$$

$$\top \sqsubseteq \leq nr.D \iff \forall x \forall y_1, \ldots \forall y_{n+1} \Big[\Big(\bigwedge_{1 \leq i \leq n+1} \Big(r(x, y_i) \wedge D(y_i) \Big) \rightarrow \bigvee_{1 \leq i < j \leq n+1} y_i = y_j \Big) \Big]$$

When using unary encoding, number restrictions actually represent first-order logic formulae of quadratic size. This is reflected in our calculus by the number of clauses that can be inferred from just a single pair of clauses. In fact, our calculus allows us to infer up to an additional exponential more clauses than the other calculi presented in the thesis.

For reasoning with number restrictions, different techniques have been proposed in the literature. In resolution based approaches, a common way is to express number restrictions using inequations between terms, as in the first-order logic representation. This approach is followed by all consequence-based reasoning methods we presented in the related work section (see Section 2.7.2), but also for example by the hypertableau algorithm used by HermiT (Motik et al., 2009). For uniform interpolation however, this approach is problematic, because it is not easy to transform clauses with inequalities back into axioms with number restrictions.

In tableau-based reasoning, a simple approach is to create n successor-nodes in the current branch for $\geq$-restrictions, and to non-deterministically merge successor-nodes for $\leq$-restrictions (see for example Baader and Sattler, 2001). If the numbers occurring in the ontology are large, this can lead to the addition of a large number of nodes to the current branch, and to a large number of branching points in the tableau. A more advanced approach, presented in Farsiniamarj and Haarslev (2010), is to encode interactions between number restrictions as inequation systems and combine the tableau-reasoner with an inequation solver. This increases the practicality if large numbers are used in the ontology. However, as we argued in Chapter 3 and 4, tableau-based methods are not well-suited for a practical computation of uniform interpolants, and it is not straightforward how these approaches can be used in a resolution-based framework.

For these reasons, instead of using an existing solution, we develop a new way of dealing with number restrictions that is compatible to the reasoning approaches we have presented so far. Our assumption is that the majority of axioms in $\mathcal{SHQ}$ ontologies do not use number restrictions, and that the numbers occurring in number restrictions are usually small. That this is a reasonable assumption is for example reflected in the NCBO BioPortal repository, a well-known repository of real-life ontologies that we discuss in more detail in Chapter 8. There are 74 ontologies in this repository that use cardinality restrictions. On average, only 2.3% of the axioms in these ontologies contain a cardinality restriction. Moreover, only 2.8% of all cardinality restrictions in the repository use a cardinality larger than 2. Out of the 5,529 cardinality restrictions that occur in the complete repository, only 14 use a cardinality that is larger than 6.

Ideally, our calculus for $\mathcal{SHQ}$ should therefore behave in a similar way as $Int_{\mathcal{ALCHF}}$ on axioms that are purely in $\mathcal{ALCHF}$, unless if entailments can be inferred that are only expressible in $\mathcal{SHQ}$. At the same time, it should generalise rules in $Int_{\mathcal{ALCHF}}$ instead of adding new rules, to avoid the resulting calculus to become too complex. The calculi presented in this chapter follow all these requirements.

6.1 The Refutation Calculus

We start by generalising the normal form to incorporate number restrictions. Every role restriction can equivalently be expressed as a number restriction, which is why the normal form for $\mathcal{SHQ}$ does not use any other types of role restrictions. At the same time, it turns out that we need more expressivity on concepts that occur under role restrictions.

Definition 6.1.1. An $\mathcal{SHQ}$ *literal* is a concept of one of the following forms:

$$A \mid \neg A \mid \geq nr.\mathcal{D} \mid \leq mr.\neg\mathcal{D},$$

where $A \in N_c$, $r \in N_r$, $n \geq 1$, $m \geq 0$ are natural numbers and $\mathcal{D} = D_1 \sqcup \ldots \sqcup D_n$ is a disjunction of definer symbols. The empty disjunction of definer symbols is expressed as $\bot$. A literal of the form $\neg D, D \in N_d$, is called *negative definer literal*. An $\mathcal{SHQ}$ *clause* is a role inclusion of the form $r \sqsubseteq s$, a transitivity axiom of the form $\mathsf{trans}(r)$, or a clause of the form $\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n$, where each L_i, $1 \leq i \leq n$, is an $\mathcal{SHQ}$ literal. An ontology is in $\mathcal{SHQ}$ *normal form* if every axiom in it is an $\mathcal{SHQ}$ clause.

Note that in contrast to the normal forms presented in former chapters, this normal form allows for disjunctions of definers under role restrictions. Even though such literals are not introduced by the normal form transformation, this flexibility is used by the rules of the calculus.

Seemingly, the normal form allows for nested negations. More precisely, under a $\leq$-restriction there is always a negated disjunction. This is necessary since $\leq$-restrictions implicitly contain a negation by themselves. This becomes apparent from their first-order logic representation (see above), and from the equivalence $\neg(\geq nr.C) \equiv \leq(n-1)r.C$. The negation in literals of the form $\leq mr.\neg\mathcal{D}$ ensures that $\mathcal{SHQ}$ ontologies in normal form correspond to formulae in negation normal form. Note that universal restrictions of the form $\forall r.D$ are represented as literals of the form $\leq 0r.\neg D$ in $\mathcal{SHQ}$ normal form.

In $Res_{\mathcal{ALC}}$ and $Res_{\mathcal{ALCHF}}$, new symbols representing conjunctions of definers are introduced dynamically. We use the same mechanism in $Res_{\mathcal{SHQ}}$. Since the normal form allows for disjunctions of definers under a role restriction, a single rule application can lead to the introduction of several definers. If $\mathcal{D}_a = D_{a1} \sqcup \ldots \sqcup D_{an}$ and $\mathcal{D}_b =$

$D_{b1} \sqcup \ldots \sqcup D_{bm}$, then $\mathcal{D}_{ab}$ represents the conjunction of $\mathcal{D}_a$ and $\mathcal{D}_b$ iff

$$\mathcal{D}_{ab} = \begin{array}{ccc} D_{a1b1}\sqcup & \ldots & \sqcup D_{a1bm}\sqcup \\ \vdots & \ddots & \vdots \\ \sqcup D_{anb1}\sqcup & \ldots & \sqcup D_{anbm} \end{array}$$

where each D_{aibj}, $1 \leq i \leq n$, $1 \leq j \leq m$, is a possibly introduced definer representing $D_{ai} \sqcap D_{bj}$. Even though this matrix looks big, we do not expect large disjunctions to occur in inferences from realistic ontologies. This is due to the same assumption we made in the introduction of this chapter, namely that only a minor part of the input ontology uses cardinality restrictions, and that only small cardinalities are used.

The rules of the refutation calculus $Res_{\mathcal{SHQ}}$ are shown in Figure 6.1. The first two rules are adaptations of corresponding rules in $Res_{\mathcal{SH}}$. The resolution rule is unchanged, and the transitivity rule is adjusted to the syntax of the $\mathcal{SHQ}$ normal form. Recall that in $\mathcal{SHQ}$, transitive roles or roles with transitive sub-roles are only allowed to occur in existential and in universal role restrictions (see Section 3.1). Due to the normal form, this means they are only to be expected in number restrictions of the form $\geq 1r.\mathcal{D}$ or $\leq 0r.\neg\mathcal{D}$. For this reason, we do not need a more general rule for transitive role axioms.

The last two rules, the $\geq$-resolution rule and the $\geq$-elimination rule, adapt the $\exists$-elimination rule to the generalised context of the $\mathcal{SHQ}$ normal form. Using the $\exists$-elimination of $Res_{\mathcal{ALC}}$, the conclusion C is inferred from the two premises $C \sqcup \exists r.D$ and $\neg D$. The $\geq$-resolution and the $\geq$-elimination rule generalise this idea to number restrictions that can contain several definer symbols. The $\geq$-resolution rule is used to eliminate unsatisfiable definers from a $\geq$-restriction. If all definers in a $\geq$-restriction have been eliminated in this way, the literal is of the form $\geq nr.\bot$, and it is eliminated using the $\geq$-elimination rule.

The two remaining rules are the $\geq$-combination rule and the $\geq\leq$-combination rule. These correspond to the role propagation rules in $Int_{\mathcal{ALCHF}}$. Before we prove their soundness in detail, we show how they relate to the rules in $Int_{\mathcal{ALCHF}}$. This may also give us some understanding on the mechanics of the calculus.

The $\forall\exists$-role propagation rule of $Res_{\mathcal{ALC}}$ has as premises two clauses $C_1 \sqcup \forall r.D_1$ and $C_2 \sqcup \exists r.D_2$ and as conclusion the clause $C_1 \sqcup C_2 \sqcup \exists r.D_{12}$. If represented in $\mathcal{SHQ}$ normal form, the premises are of the form $C_1 \sqcup \leq 0r.\neg D_1$ and $C_2 \sqcup \geq 1r.D_2$, and

Resolution:

$$\frac{C_1 \sqcup A \qquad C_2 \sqcup \neg A}{C_1 \sqcup C_2}$$

Transitivity:

$$\frac{C \sqcup \leq 0 r_1.\neg D \qquad \mathsf{trans}(r_2) \qquad r_2 \sqsubseteq_{\mathcal{N}} r_1}{C \sqcup \leq 0 r_2.\neg D' \qquad \neg D' \sqcup D \qquad \neg D' \sqcup \leq 0 r_2.\neg D'}$$

where D' is a new definer symbol.

$\geq$-Combination:

$$\frac{C_1 \sqcup \geq n_1 r_1.\mathcal{D}_1 \qquad C_2 \sqcup \geq n_2 r_2.\mathcal{D}_2 \qquad r_1 \sqsubseteq_{\mathcal{N}} r \qquad r_2 \sqsubseteq_{\mathcal{N}} r}{\begin{array}{c} C_1 \sqcup C_2 \sqcup \geq (n_1+n_2) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup \geq 1 r_1.\mathcal{D}_{12} \\ \vdots \\ C_1 \sqcup C_2 \sqcup \geq (n_1+1) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup \geq n_2 r_1.\mathcal{D}_{12} \end{array}}$$

where $\mathcal{D}_{12}$ represents $\mathcal{D}_1 \sqcap \mathcal{D}_2$.

$\geq\leq$-Combination:

$$\frac{C_1 \sqcup \geq n_1 r_1.\mathcal{D}_1 \qquad C_2 \sqcup \leq n_2 r_2.\neg\mathcal{D}_2 \qquad r_1 \sqsubseteq_{\mathcal{N}} r_2 \qquad n_1 > n_2}{C_1 \sqcup C_2 \sqcup \geq (n_1 - n_2) r_1.\mathcal{D}_{12}}$$

where $\mathcal{D}_{12}$ represents $\mathcal{D}_1 \sqcap \mathcal{D}_2$.

$\geq$-Resolution:

$$\frac{C \sqcup \geq n r.(\mathcal{D} \sqcup D) \qquad \neg D}{C \sqcup \geq n r.\mathcal{D}}$$

$\geq$-Elimination:

$$\frac{C \sqcup \geq n r.\bot}{C}$$

Figure 6.1: Inference rules of the calculus $Res_{\mathcal{SHQ}}$.

the conclusion is of the form $C_1 \sqcup C_2 \sqcup \geq 1r.D_{12}$. We can see that the $\forall\exists$-role propagation rule, translated to $\mathcal{SHQ}$ normal form, is an instance of the $\geq\leq$-combination rule, where $n_1 = 1$ and $n_2 = 0$.

Similarly, the $\exists\exists$-role propagation rule of $Res_{\mathcal{ALCHF}}$ is related to the $\geq$-combination rule. The $\exists\exists$-role propagation rule infers from the premises $C_1 \sqcup \exists R.D_1$ and $C_2 \sqcup \exists R.D_2$ the conclusion $C_1 \sqcup C_2 \sqcup \geq 2R.\top \sqcup \exists r.D_{12}$. The corresponding derivation in $Res_{\mathcal{SHQ}}$ infers from the premises $C_1 \sqcup \geq 1R.D_1$ and $C_2 \sqcup \geq 1R.D_2$ the conclusion $C_1 \sqcup C_2 \sqcup \geq 2R.(D_1 \sqcup D_2) \sqcup \geq 1R.D_{12}$. Clearly, $C_1 \sqcup C_2 \sqcup \geq 2R.(D_1 \sqcup D_2) \sqcup \geq 1R.D_{12} \models C_1 \sqcup C_2 \sqcup \geq 2R.\top \sqcup \exists R.D_{12}$. However, the conclusion of the $\geq$-combination preserves more information in the $\geq 2R$-restriction, due to the higher expressivity of $\mathcal{SHQ}$.

Finally, the two $\geq 2R.\top$-elimination rules in $Res_{\mathcal{ALCHF}}$ can be simulated in $Res_{\mathcal{SHQ}}$ in several steps.

The $\geq 2R.\top$-elimination rule I in $Res_{\mathcal{ALCHF}}$ infers from the two clauses $C_1 \sqcup \geq 2R.\top$ and $C_2 \sqcup \leq 1R.\top$ the clause $C_1 \sqcup C_2$. In $\mathcal{SHQ}$ normal form, the premises are of the following form:

1. $C_1 \sqcup \geq 2R.D_1$ *(Premise 1)*
2. $C_2 \sqcup \leq 1R.\neg D_2$ *(Premise 2)*
3. $\neg D_2$

Note that the unsatisfiable definer D_2 is necessary since we cannot express $\leq 1R.\top$ directly in $\mathcal{SHQ}$ normal form. Using the $\geq\leq$-combination rule, we infer the following clauses:

4. $C_1 \sqcup C_2 \sqcup \geq 1R.D_{12}$ *($\geq\leq$-Combination 1, 2)*
5. $\neg D_{12} \sqcup D_1$ *($D_{12} \sqsubseteq D_1$)*
6. $\neg D_{12} \sqcup D_2$ *($D_{12} \sqsubseteq D_2$)*

Since D_2 is unsatisfiable, so is D_{12}, and the last literal in Clause 4 can be eliminated using resolution, $\geq$-resolution and $\geq$-elimination.

7. $\neg D_{12}$ *(Resolution 3, 6)*
8. $C_1 \sqcup C_2 \sqcup \geq 1R.\bot$ *($\geq$-Resolution 4, 7)*
9. $C_1 \sqcup C_2$ *($\geq$-Elimination 8)*

The last clause corresponds to the conclusion of the $\geq 2R.\top$-elimination rule I.

The $\geq 2R.\top$-elimination rule II in $Res_{\mathcal{ALCHF}}$ has three clauses as premises, which are of the forms $C_1 \sqcup \geq 2R.\top$, $C_1 \sqcup \forall R.D_1$ and $\neg D_1$, and as conclusion the clause $C_1 \sqcup C_2$. We derive the same conclusion using the rules of $Res_{\mathcal{SHQ}}$, if we represent $\top$ using the definer $D_\top$:

1. $C_1 \sqcup \geq 2R.D_\top$	*(Premise 1)*
2. $C_2 \sqcup \leq 0R.\neg D_1$	*(Premise 2)*
3. $\neg D_1$	*(Premise 3)*
4. $C_1 \sqcup C_2 \sqcup \geq 2R.D_{\top 1}$	*($\geq\leq$-Combination 1, 2)*
5. $\neg D_{\top 1} \sqcup D_\top$	*($D_{\top 1} \sqsubseteq D_1$)*
6. $\neg D_{\top 1} \sqcup D_1$	*($D_{\top 1} \sqsubseteq D_1$)*
7. $\neg D_{\top 1}$	*(Resolution 3, 6)*
8. $C_1 \sqcup C_2 \sqcup \geq 2R.\bot$	*($\geq$-Resolution 4, 7)*
9. $C_1 \sqcup C_2$	*($\geq$-Elimination 8)*

Clause 9 corresponds to the conclusion of the second $\geq 2R.\top$-elimination rule of $Res_{\mathcal{ALCHF}}$.

We established that $Res_{\mathcal{SHQ}}$ is a true generalisation of $Res_{\mathcal{SH}}$ and $Res_{\mathcal{ALCHF}}$. It remains to show that the new rules are actually sound. We prove the soundness of the $\geq$-combination rule and the $\geq\leq$-combination rule in two lemmata.

Lemma 6.1.2. *The $\geq$-combination rule is sound.*

Proof. The four premises of the rule are $C_1 \sqcup \geq n_1 r_1.\mathcal{D}_1$, $C_2 \sqcup \geq n_2 r_2.\mathcal{D}_2$, $r_1 \sqsubseteq r$ and $r_2 \sqsubseteq r$. Since $((A \sqcup B) \sqcap (C \sqcup D) \models (A \sqcup C \sqcup (B \sqcap C))$, it is sufficient to prove that the following concept inclusion follows from the role inclusions $r_1 \sqsubseteq r$ and $r_2 \sqsubseteq r$, for all $0 \leq i < n_2$.

$$\geq n_1 r_1.\mathcal{D}_1 \sqcap \geq n_2 r_2.\mathcal{D}_2 \sqsubseteq \geq (n_1 + n_2 - i)r.(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup \geq (1+i)r_1.(\mathcal{D}_1 \sqcap \mathcal{D}_2) \quad (6.1)$$

Let $\mathcal{I}$ be a model of the role inclusions and $x \in \Delta^\mathcal{I}$ a domain element of the model such that $x \in (\geq n_1 r.\mathcal{D}_1)^\mathcal{I}$ and $x \in (\geq n_2 r.\mathcal{D}_2)^\mathcal{I}$. Since $\mathcal{I}$ is a model, there must be two sets of domain elements $X_1, X_2 \subseteq \Delta^\mathcal{I}$, such that the following properties are fulfilled.

1. $(x, x_i) \in r^{\mathcal{I}}$ for all $x_i \in (X_1 \cup X_2)$
2. $X_1 \subseteq \mathcal{D}_1^{\mathcal{I}}$ and $X_2 \subseteq \mathcal{D}_2^{\mathcal{I}}$
3. $\#X_1 \geq n_1$ and $\#X_2 \geq n_2$

X_1 contains the r_1-successors of x satisfying $\mathcal{D}_1$ and X_2 contains the r_2-successors of x satisfying $\mathcal{D}_2$. Without additional information, it is impossible to know how many elements are in the intersection $X_1 \cap X_2$. Suppose that we additionally have $x \in (\leq m r.(\mathcal{D}_1 \sqcup \mathcal{D}_2))^{\mathcal{I}}$, where m is any number with $m < n_1 + n_2$. Under this assumption, since $X_1 \subseteq \mathcal{D}_1^{\mathcal{I}}$ and $X_2 \subseteq \mathcal{D}_2^{\mathcal{I}}$, we can deduce that $(X_1 \cup X_2) \leq m$. This allows us to compute the minimal size of the intersection $X_1 \cap X_2$.

$$\#(X_1 \cap X_2) = \#X_1 + \#X_2 - \#(X_1 \cup X_2) \geq n_1 + n_2 - m$$

For elements $x_i \in (X_1 \cap X_2)^{\mathcal{I}}$, we have $x_i \in (\mathcal{D}_1 \sqcap \mathcal{D}_2)^{\mathcal{I}}$. Hence, we can establish that x has at least $(n_1 + n_2 - m)$ r_1-successors that satisfy $\mathcal{D}_1 \sqcap \mathcal{D}_2$, in other words, that $x \in (\geq (n_1 + n_2 - m) r_1.(\mathcal{D}_1 \sqcap \mathcal{D}_2))^{\mathcal{I}}$. Since this was deduced from the assumption that $x \in (\leq m r.(\mathcal{D}_1 \sqcup \mathcal{D}_2))^{\mathcal{I}}$, we can conclude that in every model $\mathcal{I}$ in which $x \in (\geq n_1 r_1.\mathcal{D}_1)^{\mathcal{I}}$, $x \in (\geq n_2 r_2.\mathcal{D}_2)^{\mathcal{I}}$, $\mathcal{I} \models r_1 \sqsubseteq r$ and $\mathcal{I} \models r_2 \sqsubseteq r$, x satisfies the following concept.

$$\neg(\leq m r.(\mathcal{D}_1 \sqcup \mathcal{D}_2)) \sqcup \geq (n_1 + n_2 - m) r_1.(\mathcal{D}_1 \sqcap \mathcal{D}_2) \tag{6.2}$$

This concept is equivalent to the following.

$$\geq (m+1) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup \geq (n_1 + n_2 - m) r_1.(\mathcal{D}_1 \sqcap \mathcal{D}_2) \tag{6.3}$$

Since $m < n_1 + n_2$, we can set $m = n_1 + n_2 - i - 1$, where $1 \leq i < n_1 + n_2$. Concept (6.3) can then be rewritten in the following way using i.

$$\geq (n_1 + n_2 - i) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup \geq (1 + i) r_1.(\mathcal{D}_1 \sqcap \mathcal{D}_2) \tag{6.4}$$

We established that if x satisfies $(\geq n_1 r_1.\mathcal{D}_1 \sqcap \geq n_2 r_2.\mathcal{D}_2)$, it also satisfies Concept (6.4). Therefore, Entailment (6.1) follows from the role inclusions, and the $\geq$-combination rule is sound. □

Lemma 6.1.3. *The $\geq\leq$-combination rule is sound.*

Proof. Observe that from the premises $C_2 \sqcup \leq n_2 r_2.\neg\mathcal{D}_2$ and $r_1 \sqsubseteq_{\mathcal{N}} r_2$ of the rule, $C_2 \sqcup \leq n_2 r_1.\neg\mathcal{D}_2$ can be deduced. It is therefore sufficient to consider the case where the premises of the rule use the same role.

To validate soundness of the $\geq\leq$-rule, we have to prove that the following concept inclusion is valid for any $n_2 < n_1$.

$$\geq n_1 r.\mathcal{D}_1 \sqcap \leq n_2 r.\neg\mathcal{D}_2 \sqsubseteq \geq (n_1 - n_2) r.(\mathcal{D}_1 \sqcap \mathcal{D}_2) \tag{6.5}$$

Let $\mathcal{I}$ be a model with a domain element $x \in \Delta^{\mathcal{I}}$ that satisfies the concept on the left-hand side of the entailment. We have $x \in (\geq n_1 r.\mathcal{D}_1 \sqcap \leq n_2 r.\neg\mathcal{D}_2)^{\mathcal{I}}$. Let X be the set of all r-successors of x, that is $X = \{x_i \mid (x, x_i) \in r^{\mathcal{I}}\}$. Let $X_1 = X \cap \mathcal{D}_1^{\mathcal{I}}$ contain the r-successors of x that satisfy $\mathcal{D}_1$, and $X_2 = X \cap \mathcal{D}_2^{\mathcal{I}}$ contain the r-successors of x that satisfy $\mathcal{D}_2$. Since $x \in (\geq n_1 r.\mathcal{D}_1)^{\mathcal{I}}$, $\#X_1 \geq n_1$. Furthermore, since $x \in (\leq n_2 r.\neg\mathcal{D}_2)^{\mathcal{I}}$, $\#(X \setminus X_2) \leq n_2$.

Since $n_1 \geq n_2$, X_1 and X_2 have to overlap. In addition, since $X_1 \subseteq X$ and $\#(X \setminus X_2) \leq n_2$, also $\#(X_1 \setminus X_2) \leq n_2$. This way, we obtain a lower bound for the number of elements in $X_1 \cap X_2$.

$$\#(X_1 \cap X_2) = \#X_1 - \#(X_1 \setminus X_2) \geq n_1 - n_2 \tag{6.6}$$

We have $X_1 \cap X_2 \subseteq (\mathcal{D}_1 \sqcap \mathcal{D}_2)^{\mathcal{I}}$, and can establish that x has at least $n_1 - n_2$ r-successors that satisfy $\mathcal{D}_1 \sqcap \mathcal{D}_2$. Therefore, $x \in (\geq (n_1 - n_2) r.(\mathcal{D}_1 \sqcap \mathcal{D}_2))^{\mathcal{I}}$. Hence, Entailment 6.5 is valid, and the $\geq\leq$-combination rule is sound. □

These lemmata, together with the earlier observations concerning the other rules, allow us to establish soundness of the calculus.

Theorem 6.1.4. *$Res_{\mathcal{SHQ}}$ is sound.*

However, $Res_{\mathcal{SHQ}}$ is not terminating, since the $\geq$-rule can be applied arbitrary often, such that clauses with increasing cardinalities in $\geq$-restrictions are inferred. In order to obtain a terminating calculus, we extend $Res_{\mathcal{SHQ}}$ with redundancy elimination techniques, for which adapt the redundancy notions of $Res^{s}_{\mathcal{ALCH}}$ defined in Definition 5.1.5.

Definition 6.1.5. A *definer symbol D_1 subsumes a definer symbol D_2*, in symbols $D_1 \sqsubseteq_d D_2$, if either $D_1 = D_2$ or there is a clause $\neg D_1 \sqcup D_2$ in the current clause set.

A *disjunction $\mathcal{D}_1$ of definer symbols subsumes a disjunction $\mathcal{D}_2$ of definer symbols*, in symbols $\mathcal{D}_1 \sqsubseteq_d \mathcal{D}_2$, if every definer symbol in $\mathcal{D}_1$ subsumes a definer symbol in $\mathcal{D}_2$. A *literal L_1 subsumes a literal L_2*, in symbols $L_1 \sqsubseteq_l L_2$, if one of the following conditions hold:

1. $L_1 = L_2$
2. $L_1 = \geq n_1 r.\mathcal{D}_1$, $L_2 = \geq n_2 s.\mathcal{D}_2$, $r \sqsubseteq_{\mathcal{N}} s$, $n_1 \geq n_2$, and $\mathcal{D}_1 \sqsubseteq_d \mathcal{D}_2$.
3. $L_1 = \leq n_1 r.\neg\mathcal{D}_1$, $L_2 = \leq n_2 s.\neg\mathcal{D}_2$, $s \sqsubseteq_{\mathcal{N}} r$, $n_1 \leq n_2$ and $\mathcal{D}_1 \sqsubseteq_d \mathcal{D}_2$.

A clause C_1 *subsumes a clause* C_2 ($C_1 \sqsubseteq_c C_2$) if every literal $L_1 \in C_1$ subsumes a literal $L_2 \in C_2$. A clause C is *redundant* with respect to a clause set $\mathcal{N}$, if $\mathcal{N}$ contains a clause C' with $C' \sqsubseteq_c C$. The *reduction of a clause* C, denoted by *red*(C), is obtained from C by removing every literal that subsumes another literal in C.

In addition to the notion of redundancy used in $Res^s_{\mathcal{ALC}}$, as defined in Definition 4.3.6, this definition takes into account disjunctions of definers and numbers that occur in number restrictions. The redundancy elimination rules are the same as for $Res^s_{\mathcal{ALC}}$. With these techniques, we obtain termination for $Res^s_{\mathcal{SHQ}}$.

Theorem 6.1.6. *$Res^s_{\mathcal{SHQ}}$ is terminating.*

Proof. We already established that at most 2^n definer symbols are introduced by the calculus, where n is the number of definer symbols occurring in the initial set of clauses (see Lemma 4.2.4 and note that the way new definers are introduced is the same as in $Res_{\mathcal{ALC}}$). Since the normal form does not allow for nested disjunctions under number restrictions, this also gives us a finite bound on the number of definer disjunctions occurring in the clause set. (Recall that both clauses and disjunctions of definer symbols are represented as sets. Therefore, they cannot have duplicate elements.)

The $\geq$-combination rule is the only rule where a number occurring in the conclusion can be larger than the numbers in the premises. Suppose the $\geq$-combination rule can be applied on two clauses $C_1 \sqcup \geq n_1 r_1.\mathcal{D}_1$ and $C_2 \sqcup \geq n_2 r_2.\mathcal{D}_2$. There are two possibilities: (i) $\mathcal{D}_1 \neq \mathcal{D}_2$ or $r_1 \neq r_2$, and (ii) $\mathcal{D}_1 = \mathcal{D}_2$ and $r_1 = r_2$. Only finitely many variations of the first case are possible, since both the number of possible definer disjunctions and the number of role symbols in the RBox is bounded. Therefore, we can restrict

our attention to the second case. If $\mathcal{D}_1 = \mathcal{D}_2$ and $r_1 = r_2$, our conclusions are of the following form:

$$C_1 \sqcup C_2 \sqcup \geq(n_1 + n_2)r_2.\mathcal{D}_2 \sqcup \geq 1 r_2.\mathcal{D}_2$$
$$\vdots$$
$$C_1 \sqcup C_2 \sqcup \geq(n_1 + 1)r_2.\mathcal{D}_2 \sqcup \geq n_2 r_2.\mathcal{D}_2$$

These clauses are all subsumed by the premise $C_2 \sqcup \geq n_2 r_2.\mathcal{D}_2$. Therefore, if subsumption deletion is applied eagerly, the $\geq$-combination rule only produces new clauses up to a certain bound.

We obtain that $Res^s_{\mathcal{SHQ}}(\mathcal{N})$ is finitely bounded for any input clause set $\mathcal{N}$, and that $Res^s_{\mathcal{SHQ}}$ is terminating. □

6.2 Refutational Completeness of $Res_{\mathcal{SHQ}}$

In order to prove refutational completeness of $Res^s_{\mathcal{SHQ}}$, we adapt the model construction first introduced in Section 4.2.3 to our new calculus.

Let $\mathcal{N}$ be any set of $\mathcal{SHQ}$ clauses with $\bot \notin Res_{\mathcal{SHQ}}(\mathcal{N})$, and $\mathcal{N}^* = Res_{\mathcal{SHQ}}(\mathcal{N})$.

Let $\prec_d$, $\prec_c$ and $\prec_r$ be orderings fulfilling the same constraints as the corresponding orderings in the model construction for $Res_{\mathcal{ALCHF}}$ (see Section 5.4), that is, $\prec_c$ is an arbitrary ordering on concept symbols, $\prec_r$ is the non-reflexive sub-relation of $\sqsubseteq_{\mathcal{N}}$, and $D_1 \prec_d D_2$ for all $\neg D_1 \sqcup D_2 \in \mathcal{N}^*$.

A definer D is *maximal* in a disjunction $\mathcal{D}$ of definers if D is maximal in $\mathcal{D}$ according to $\prec_d$. $\prec_d$ is extended to a total ordering on disjunctions of definers based on the multiset extension $(\prec_d)_{mul}$.

We define a total ordering $\prec_l$ on literals based on the ordering we used in Section 5.4. Specifically, $\prec_l$ is any ordering that satisfies the following constraints, where disjunctions of definer symbols are treated as sets.

1. $A \prec_l \neg A$ for all $A \in N_c$.

2. If $A_1 \prec_c A_2$, then $\neg A_1 \prec_l A_2$

3. $\geq n_1 r.\mathcal{D}_1 \prec_l \leq n_2 r.\mathcal{D}_2$ for all $n_1 > 0$, $n_2 \geq 1$, $r \in N_r$, $\mathcal{D}_1, \mathcal{D}_2 \in 2^{N_d}$.

4. $\neg D \prec_l A$ and $\neg D \prec_l \exists r.\mathcal{D}$, for all $D \in N_d$, $\mathcal{D} \in 2^{N_d}$, $A \in N_c \setminus N_d$ and $r \in N_r$.

5. If $r \prec_r s$, $\geq n_1 r.\mathcal{D}_1 \prec_l \geq n_2 s.\mathcal{D}_2$ and $\leq n_1 r.\mathcal{D}_1 \prec_l \leq n_2 s.\mathcal{D}_2$ for all $n_1, n_2 \geq 0$ and $D_1, D_2 \in 2^{N_d}$.

6. If $D_1 \prec_d D_2$, then $\neg D_1 \prec_l D_2$

7. If $\mathcal{D}_1 \prec_d \mathcal{D}_2$, then $\geq n_1 r.\mathcal{D}_1 \prec_l \geq n_2 s.\mathcal{D}_2$ and $\leq n_1 r.\mathcal{D}_1 \prec_l \leq n_2 s.\mathcal{D}_2$ for all n_1, $n_2 \geq 0$ and $r, s \in N_r$ such that r and s are not related by $\prec_r$.

These constraints are an adaptation of the constraints given on page 128, which are used in the model construction for $Res_{\mathcal{ALCHF}}$ and $Res_{\mathcal{SIF}}$, to $\mathcal{SHQ}$ literals. That such an ordering always exist, can be shown by adaptations of the proof for Lemma 4.2.9 using a lexicographical ordering.

As in the model construction in Section 4.2.3, $\prec_l$ is extended to an ordering $\prec_c$ on clauses using the multiset extension, and the clauses in $\mathcal{N}^*$ are enumerated based on this ordering, where we denote the position of each clause $C_i \in \mathcal{N}^*$ in this enumeration by an index i. We therefore have that $C_i \prec_c C_j$ implies $i < j$.

Due to the $\geq$-number restrictions, domain elements can be connected to arbitrary numbers of successors satisfying the same definer. To take this into account, we represent model fragments using multisets.

Definition 6.2.1. *Additive literals* are of the form A or $+1r.D$, where $A \in N_c$, $r \in N_r$, $D \in N_d$. A *multiset model fragment* is a multiset of additive literals. A multiset model fragment I *satisfies a literal* L, written $I \models L$, iff

1. L is a positive literal of the form A and $A \in I$,

2. L is a negative literal of the form $\neg A$ and $A \notin I$,

3. L is of the form $\geq nr.\mathcal{D}$ and $\#\{+1s.D \in I \mid s \sqsubseteq_{\mathcal{N}} r, D \sqsubseteq_d \mathcal{D}\} \geq n$.

4. L is of the form $\leq nr.\neg\mathcal{D}$ and $\#\{+1s.D \in I \mid s \sqsubseteq_{\mathcal{N}} r, D' \not\sqsubseteq_d \mathcal{D}\} \leq n$.

A model fragment I *satisfies a clause* C, written $I \models C$, if there is a literal $L \in C$ such that $I \models L$.

Inspired by the model constructions discussed in the previous sections, for each definer D that occurs in $\mathcal{N}^*$, a multiset model fragment I^D is constructed. If $\neg D \in \mathcal{N}^*$, we again set $\mathcal{I}^D = \emptyset$. For all other definers, I^D is defined incrementally as follows.

1. $I_0^D = \{D\}$ if $D \neq \epsilon$ and $I_0^D = \emptyset$ if $D = \epsilon$.
2. If $I_{i-1}^D \models C_i$, then $I_i^D = I_{i-1}^D$, otherwise
   (a) $I_i^D = I_{i-1}^D \cup \{L\}$, if $I_{i-1}^D \not\models C_i$ and the maximal literal L in C_i is an unnegated concept symbol,
   (b) $I_i^D = I_{i-1}^D \cup \{(+1s.D')^n \mid r \sqsubseteq_{\mathcal{N}} s\}$, if $I_{i-1}^D \not\models C_i$ and the maximal literal in C_i is of the form $\geq n' r.\mathcal{D}$, where D' is the largest disjunct in $\mathcal{D}$ and n is the smallest number such that $I_{i-1}^D \cup \{(+1s.D')^n \mid r \sqsubseteq_{\mathcal{N}} s\} \models C_i$.
3. $I^D = I_n^D$, where n is the number of clauses in $d^e(D)$.

The model construction is very similar to the one used to prove refutational completeness of the other calculi in the thesis. Note that in Step 2b we always add the smallest number of additive literals necessary to satisfy the current clause.

A step in the model construction is called *effective* if the step ensures that $I_i^D \models C_i$ and $I^D \models C_i$. It is easy to verify that Steps 2a and 2b of the model construction are effective. Following Step 2b, one can prove further properties of the created multiset model fragments.

Lemma 6.2.2. *I^D satisfies the following properties.*

1. *For every $+1r.D' \in I^D$ and s with $r \sqsubseteq_{\mathcal{N}} s$, $+1s.D' \in I^D$.*
2. *If $I^D \models \geq nr.\mathcal{D}$, then also $I^D \models \geq ns.\mathcal{D}$ for all s with $r \sqsubseteq_{\mathcal{N}} s$.*
3. *If $I^D \models \leq nr.\neg\mathcal{D}$, then also $I^D \models \leq ns.\neg\mathcal{D}$ for all s with $s \sqsubseteq_{\mathcal{N}} r$.*
4. *If $I_{i-1}^D \not\models C_i$ and the maximal literal in C_i is $\geq nr.\mathcal{D}$, then $\#\{+1r.D \in I_{i+1}^D \mid \{D\} \sqsubseteq_d \mathcal{D}\} = n$.*

Proof. The first property holds since, if an additive literal of the form $+1r.D'$ is added in Step 2b, we further add an additional literal of the form $+1s.D'$ for every $r \sqsubseteq_{\mathcal{N}} s$. The other properties are stepwise consequences of this property. □

Before we prove $I^D \models C$ for all definers D and all clauses $C \in \mathcal{N}^*$, we consider a few special cases. We start by proving that conclusions of the $\geq$-combination rule do not directly contribute to additional additive literals in the model fragments.

Lemma 6.2.3. *Assume $\mathcal{N}^*$ contains two clauses of the form $C_{j_i} = C_{j_i} \sqcup \geq n_i r_i.\mathcal{D}_i$, for $i \in \{1,2\}$, where $\geq n_i r_i.\mathcal{D}_i$ is maximal in C_{j_i}, and there is a role r with $r_i \sqsubseteq_{\mathcal{N}} r$ for both i. Let C_j be a conclusion of applying the $\geq$-combination rule on C_{j_1} and C_{j_2} on the maximal literals. Then, $I^D_{j-1} \models C_j$.*

Proof. From the definition of the $\geq$-rule, it follows that C_j must be of the following form:

$$C_1 \sqcup C_2 \sqcup \geq (n_1 + n_2 - d) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup \geq (1+d) r_1.\mathcal{D}_{12}, \tag{6.7}$$

where $0 \leq x < n_2$ and $\mathcal{D}_{12}$ represents the conjunction of $\mathcal{D}_1$ and $\mathcal{D}_2$. We denote the last two literals of C_j respectively by L_1 and L_2, that is, $L_1 = \geq (n_1+n_2-d) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2)$ and $L_2 = \geq (1+d) r_1.\mathcal{D}_{12}$ The case where $I^D_{j_i} \models C_1 \sqcup C_2$ is trivial, we therefore assume $I^D_{j_i} \not\models C_1 \sqcup C_2$ for $i \in \{1,2\}$, and prove that then, $I^D_{j-1} \models L_1 \sqcup L_2$.

First observe that $C_{j_i} \prec_c C_j$ for both $i \in \{1,2\}$, since the maximal literal in C_j is of the form $\geq n r.(\mathcal{D}_1 \sqcup \mathcal{D}_2)$, and $r_{j_i} \sqsubseteq_{\mathcal{N}} r$ and $\mathcal{D}_i \sqsubseteq_d \mathcal{D}_1 \sqcup \mathcal{D}_2$ for $i \in \{1,2\}$. For the indices, this implies $j_i < j$ for each $i \in \{1,2\}$. Due to Properties 4 and 2 in Lemma 6.2.2, we therefore have $I^D_{j-1} \models \geq n_1 r.\mathcal{D}_1$ and $I^D_{j-1} \models \geq n_2 r.\mathcal{D}_2$. Denote the sets of additive literals in I^D_{j-1} that satisfy these literals by $I^D_{+\mathcal{D}_i}$:

$$I^D_{+\mathcal{D}_i} = \{+1s.D' \in I^D_j \mid \{D'\} \sqsubseteq \mathcal{D}_i, s \sqsubseteq_{\mathcal{N}} r_i\} \text{ for } i \in \{1,2\}.$$

By definition, (i) $I^D_{+\mathcal{D}_i} \models \geq n_i r_i.\mathcal{D}_i$ and (ii) $I^D_{+\mathcal{D}_i} \subseteq I^D_{j-1}$. Due to (i), we additionally have (iii) $\#I^D_{+\mathcal{D}_i} \geq n_i$ and (iv) $\{D'\} \sqsubseteq \mathcal{D}_i$ and $s \sqsubseteq_{\mathcal{N}} r_i$ for every D' with $+1s.D' \in I^D_{+\mathcal{D}_i}$.

If $I^D_{+\mathcal{D}_1} \cap I^D_{+\mathcal{D}_2} = \emptyset$, due to (iii), we have $\#(I^D_{+\mathcal{D}_1} \cup I^D_{+\mathcal{D}_2}) \geq (n_1+n_2)$. Together with (ii) and (iv), from this it follows that $I^D_{j-1} \models \geq (n_1+n_2) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2)$, which ensures $I^D_{j-1} \models L_1$ and $I^D_{j-1} \models C_j$.

Assume otherwise that $I^D_{+\mathcal{D}_1} \cap I^D_{+\mathcal{D}_2} \neq \emptyset$. Then let $I^D_{\sqcap} = I^D_{+\mathcal{D}_1} \cap I^D_{+\mathcal{D}_2}$. Because of (iv), $\{D'\} \sqsubseteq_d \mathcal{D}_1$, $\{D'\} \sqsubseteq_d \mathcal{D}_2$ and $s \sqsubseteq r_1$ for all $+1s.D' \in I^D_{\sqcap}$. If $\mathcal{D}_{12}$ represents the conjunction of $\mathcal{D}_1$ and $\mathcal{D}_2$, it can be shown that this implies $\{D'\} \sqsubseteq_d \mathcal{D}_{12}$ for every D' with $+1s.D' \in I^D_{\sqcap}$. Consequently, $I^D_{\sqcap} \models \geq n_{\sqcap} r_1.\mathcal{D}_{12}$, where $n_{\sqcap} = \#I^D_{\sqcap}$. With (ii), this gives us (v) $I^D_{j-1} \models \geq n_{\sqcap} r_1.\mathcal{D}_{12}$ and (vi) $I^D_{j-1} \models \geq (n_1 + n_2 - n_{\sqcap}) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2)$.

There are two possibilities regarding the number d in Equation 6.7.

1. $n_{\sqcap} \leq d$. We then have that $I^D_{j-1} \models L_1$ due to (vi).

2. $n_{\sqcap} > d$. In this case, $I^D_{j-1} \models L_2$, due to (v).

We have established that either way, $I^D_{j-1} \models C_j$. □

Next, we prove that for a specific set of saturated clauses, the corresponding model fragments satisfy unary clauses of the form $\leq nr.\mathcal{D}$.

Lemma 6.2.4. *Assume we have a satisfiable set of clauses* $\mathcal{N} = \{\leq nr.\neg D_a\} \cup \{\geq n_i r_i.D_i \mid 1 \leq i \leq m, r_i \sqsubseteq_{\mathcal{N}} r\} \cup \mathcal{M}$*, where* $\mathcal{M}$ *only consists of clauses with at least one negative definer literal. Let* $\mathcal{N}^* = Res_{\mathcal{SHQ}}(\mathcal{N})$*, and* I^ϵ *be the model fragment for* ϵ *generated based on the clauses in* $\mathcal{N}^*$*. Then,* $I^\epsilon \models \leq nr.\neg D_a$*.*

Proof. We have to show that $\#\{+1r_i.D_i \in I^\epsilon \mid r_i \sqsubseteq r, \neg D_i \sqcup D_a \notin \mathcal{N}^*\} \leq n$.

If $\Sigma_{i=1}^m n_i < n$, at most $\Sigma_{i=1}^m n_i$ additive literals are added to I^ϵ for the clauses $\mathcal{N}$ and due to Lemma 6.2.3, no further additive literals are added for clauses that can be derived from $\mathcal{N}$. We obtain that in this case, $I^\epsilon \models \leq nr.\neg D_a$.

Otherwise, set $n^\Delta = \Sigma_{i=1}^m n_i - n$. The proof is by induction on the clauses in $\mathcal{N}$, where the base case is that $n^\Delta \leq n_i$ for all clauses $\geq n_i r_i.D_i \in \mathcal{N}$.

Base case: Assume $n^\Delta < n_i$ for all $\geq n_i r_i.D_i \in \mathcal{N}$. Applying the $\geq$-rule incrementally on each clause in $\mathcal{N}$ yields a set of clauses of the form $\geq n^\Sigma r^\Sigma.\mathcal{D}^\Sigma \sqcup C^\Sigma_k$, where $n^\Sigma = \Sigma_{i=1}^m n_i$, $r^\Sigma \sqsubseteq_{\mathcal{N}} r$, $\mathcal{D} = \{D_i \mid 0 < i \leq m\}$. Each C^Σ_k consists of number restrictions of the form $\geq n' r'.\mathcal{D}'$, $n' \leq n_i, i \leq m$, where $\mathcal{D}'$ contains definers representing conjunctions of definers in $\mathcal{N}$.

Applying the $\geq\leq$-rule with on this clause and $\leq nr.\neg D_a$ yields clauses of the form $C^\Delta_k = \geq n^\Delta r^\Sigma.\mathcal{D}^\Delta \sqcup C^\Sigma_k$, where $\mathcal{D}^\Delta = \{D_{ia} \mid 1 \leq i \leq m\}$ consists of definers D_{ia} representing $D_i \sqcap D_a$ for $1 \leq i \leq m$. Every number restriction in C^Δ_k contains definers that represent conjunctions of definers in $\mathcal{N}$. Therefore, each C^Δ_k is smaller than the clauses $\geq n_i r_i.D_i \in \mathcal{N}$ and processed before these clauses by the construction of I^ϵ. Due to Lemma 6.2.3, we can also ignore saturations on the clauses of C^Δ_k via the $\geq$-combination rule.

For clauses of the form C_k^Δ, in Step 2b of the construction of I^ϵ, we add n^Δ additive literals of the form $+1r.D_{ij}$, where each D_{ij} subsumes at least two definer symbols from $\mathcal{N}$. Since C^Δ is smaller than the clauses in $\mathcal{N}$, for each of these additive literals where $\neg D_{ij} \sqcup D_a \notin \mathcal{N}^*$, 2 additive definer literals less are added when the $\geq$-restrictions in $\mathcal{N}$ are processed. For each of these additive literals where $\neg D_{ij} \sqcup D_a \in \mathcal{N}^*$, 1 additive definer literal less is added when $\geq$-restrictions in $\mathcal{N}$ are processed. As a result, only $n^\Sigma - n^\Delta = n$ definer literals of the form $+1r'.D'$, where $\neg D' \sqcup D_a \notin \mathcal{N}^*$, are added to the multiset model fragment, and hence, we obtain that $\#\{+1r_i.D_i \in I^\epsilon \mid r_i \sqsubseteq r, D_i \not\sqsubseteq D_a\} = n$ and $I^\epsilon \models \leq nr.\neg D_a$.

Induction Step: Let $\mathcal{N}_1$ be a clause set for which the inductive hypothesis holds, and $\geq n_2 r.D_2$ be a clause not in $\mathcal{N}_1$. We show that the hypothesis holds for $\mathcal{N} = \mathcal{N}_1 \cup \{\geq n_2 r.D_2\}$. Without loss of generality, we assume $D' \prec_d D_2$ for all definers occurring in $\mathcal{N}_1$.

Let $\mathcal{N}_1^*$ be the saturation of $\mathcal{N}_1$ using $Res_{\mathcal{SHQ}}$. As in the base case, there is a clause $C_1^\Sigma = \geq n^\Sigma r^\Sigma.\mathcal{D}^\Sigma \sqcup \ldots \in \mathcal{N}_1^*$ which is the result of incrementally applying the $\geq$-combination rule on all $\geq$-restrictions in $\mathcal{N}_1$, and a clause $C_1^\Delta = \geq n^\Delta r.\mathcal{D}^\Delta \sqcup C_1^{\Delta\prime} \in \mathcal{N}_1^*$, which is the result of applying the $\geq\leq$-combination rule on this clause and $\leq nr.\neg D_a$. Applying the $\geq$-combination rule on C_1^Σ and $\geq n_2 r.D_2$ results in a new clause $C_2^\Sigma = \geq (n^\Sigma + n_2)r.(\mathcal{D} \sqcup D_2) \sqcup C_1^{\Delta\prime}$, and applying the $\geq\leq$-rule on this clause results in a clause $C_2^\Delta = \geq (n^\Delta + n_2)r.(\mathcal{D}^\Delta \sqcup D_{2a}) \sqcup C_1^{\Delta\prime}$. Since $D' \prec_d D_2$ for all definer symbols in $\mathcal{N}_1$, we also have that D_{2a} is larger than all definer symbols in $\mathcal{D}^\Delta$, which represent conjunctions of definers from $\mathcal{N}_1$. Therefore, $C_1^\Delta \prec_c C_2^\Delta$. If $\neg D_{2a} \in \mathcal{N}^*$, $\mathcal{N}^*$ additionally contains conclusions of the $\geq$-resolution rule with $\neg D_{2a}$.

1. Assume $\neg D_{2a} \notin \mathcal{N}^*$, which means that the $\geq$-resolution rule is not applicable on D_{2a}. Since $C_1^\Delta \prec_c C_2^\Delta$, C_2^Δ is processed after C_1^Δ by the model construction, but still before $\geq n_2 r.D_2$. $\geq (n^\Delta + n_2)r.(\mathcal{D}^\Delta \sqcup D_{2a})$ is the maximal literal in C_2^Δ, and D_{2a} is larger than all definers in $\mathcal{D}^\Delta$. Therefore, in Step 4 of the model construction, n_2 additive literals of the form $+1r.D_{2a}$ are added to I^ϵ. $\neg D_{2a} \sqcup D_2 \in \mathcal{N}^*$, therefore, when $\geq n_2 r.D_2$ is processed, no additive

literals of the form $+1r.D_2$ are added. We also have $\neg D_{2a} \sqcup D_a \in \mathcal{N}^*$, and due to the inductive hypothesis, $\#\{+1r_i.D_i \in I^\epsilon \mid r_i \sqsubseteq_{\mathcal{N}} r, D_i \not\sqsubseteq D_a\} < n$, and hence $I^D \models \leq nr.\neg D_a$.

2. If $\neg D_{2a} \in \mathcal{N}^*$, then applying the $\geq$-resolution rule on the clauses $\neg D_{2a}$ and $\geq(n^\Delta + n_2)r.(\mathcal{D}^\Delta \sqcup D_{2a}) \sqcup C_a^{\Sigma'}$ results in a set of smaller clauses of the form $C_a^\Delta = \geq(n^\Delta + n_2)r.(\mathcal{D}^\Delta) \sqcup C_a^{\Sigma'}$. But $C_1^{\Sigma'}$ contains a literal $\geq n_2 r.\mathcal{D}_{2i}$, where $\mathcal{N}^* \models \mathcal{D}_{2i} \sqsubseteq D_2$ and $\mathcal{N}^* \models \mathcal{D}_{2i} \sqsubseteq \mathcal{D}_i$ for at least one disjunction of definers $\mathcal{D}_i$ in $\mathcal{N}_1^*$. Since $D_j \prec_d D_2$ for all definer symbols D_j in $\mathcal{N}_1$, and since every definer $D_{jk} \neq D_{2i}$ in $C_a^{\Sigma'}$ represents the disjunctions over conjunctions of definers that are smaller than D_2, $\geq n_2 r.\mathcal{D}_{2i}$ is the maximal literal in C_a^Δ. As a result, n_2 literals of the form $+1r.D_{2i}$, $D_{2i} \in \mathcal{D}_{2i}$, are added in Step 4 of the model construction. Since $\geq n_2 r.\mathcal{D}_{2i}$ is maximal in C_a^Δ, all clauses in $\mathcal{N}_1^*$ where $\geq n'r'.\mathcal{D}_i$ is maximal are larger than C_a^Δ, and therefore processed subsequently by the model construction. As a result, n_2 additive literals less are added for these, than it would be the case for the clause set $\mathcal{N}_1^*$. Also, no additive literals are added for the clause $\geq n_2 r.D_2$. By the inductive hypothesis, for $\mathcal{N}_1^*$, at most n additive literals of the form $+1r_i.D_i, r_i \sqsubseteq r, \neg D_i \sqcup D_a \in \mathcal{N}^*$ are part of the model fragment. For $\mathcal{N}$, we add n_2 additive literals of this form less when processing the clauses in $\mathcal{N}_1^*$, and we add n_2 additive literals of the form $+1r.\mathcal{D}_{2i}$. Therefore, we have again that $\#\{+1r_i.D_i \in I^\epsilon \mid r_i \sqsubseteq r, D_i \not\sqsubseteq D_a\} \leq n$, and hence $I^\epsilon \models \leq nr.D_a$.

We have shown that $I^\epsilon \models \leq nr.D_a$ for any set of clauses that satisfy the conditions of the lemma. □

Lemma 6.2.4 can be generalised to the following lemma.

Lemma 6.2.5. *Let $\mathcal{N}^*$ be a saturated set of clauses and $C_i = C_i' \sqcup \leq nr.D \in \mathcal{N}^*$ be a clause in which $\leq nr.D'$ is maximal. Let I^D be the model fragment for a definer D generated from $\mathcal{N}^*$. Then, $I^D \models C_i$.*

Proof. Assume $I^D \not\models C_i$. Since $\leq nr.D' \in C_i$, we also have $I^D \not\models \leq nr.D'$. This can only be the case due to clauses of the form $C_{i_j} = \geq n_j r_j.D_j \sqcup C_{i_j}{}'$, $r_j \sqsubseteq r$, $\neg D_j \sqcup D_i \in \mathcal{N}^*$ where $\geq n_j r_j.D_j$ is maximal, and $I_{j-1}^D \not\models C_j$. Let $\mathcal{N}'$ be a subset of $\mathcal{N}'$ that only

contains these clauses, C_i and any unary clause of the form $\neg D' \in \mathcal{N}^*$, and let $\mathcal{N}'^*$ be the saturation of $\mathcal{N}'$. Let I'^D be the model fragment for D generated from $\mathcal{N}'^*$. $\mathcal{N}'^*$ contains all clauses from $\mathcal{N}^*$ that are crucial for $I^D \not\models \leq nr.D'$, and therefore $I'^D \not\models \leq nr.D'$. Let $\mathcal{N}''$ be a clause set that contains a unary clause C_{max} for each clause $C \in \mathcal{N}'$, where C_{max} contains the maximal literal in C. Let I''^D be the model fragment for D generated from the saturation $\mathcal{N}''^*$. Since $I'^D \not\models \leq nr.D'$, and I'^D is solely built based on the maximal literals in each clause, $I''^D \not\models \leq nr.D'$. On the other hand, $\mathcal{N}''$ is a clause set of the same form as in Lemma 6.2.4, which implies $I''^D \models \leq nr.D'$. We have a contradiction, which means the initial assumption $I^D \not\models C_i$ cannot be true. Hence, $I^D \models C_i$. □

We have now everything we need to prove that each non-empty model fragment I^D satisfies all clauses in $\mathcal{N}^*$.

Lemma 6.2.6. *Let D be any definer with $\neg D \notin \mathcal{N}^*$ or $D = \epsilon$. Then, $I^D \models C$ for all clauses $C \in \mathcal{N}^*$.*

Proof. We validate that for each $C_i \in \mathcal{N}^*$, $I^D \models C_i$. The proof is by contradiction. Assume i is the smallest i with $I^D \not\models C_i$.

1. The maximal literal in C_i is of the form A or $\geq nr.D'$. Then, $I^D \models C_i$ by construction of the model, which contradicts the assumption that $I^D \not\models C_i$.

2. If the maximal literal in C_i is of the form $\neg A$, $I^D \not\models \neg A$ and therefore $A \in I^D$. Then the situation is the same as for $Res_{\mathcal{ALC}}$, and we have a contradiction for the same reasons as in the proof for Lemma 4.2.15.

3. If the maximal literal in C_i is of the form $\leq nr.\neg D_1$, by Lemma 6.2.5, $I^D \models C_i$, which contradicts the initial assumption that $I^D \not\models C_i$.

We have shown that all cases lead to a contradiction, which means that there cannot be a smallest clause $C \in \mathcal{N}^*$ such that $I^D \not\models C$. Hence, $I^D \models C$ for all clauses $C \in \mathcal{N}^*$. □

Based on the models fragments, we build a candidate model inspired by the model construction we used to prove refutational completeness of $Res_{\mathcal{ALC}}$ in Section 4.2.3. However, as for $Res_{\mathcal{ALCHF}}$, it is not sufficient to just have one domain element per

definer symbol, since our candidate model has to be able to satisfy concepts of the form $\geq nr.D$. Let n_{max} be the largest number n occurring in a number restriction $\geq nr.D$ in $\mathcal{N}^*$. For each definer D, the domain $\Delta^{\mathcal{I}}$ of the candidate model $\mathcal{I}$ contains n_{max} domain elements x_i^D, where $1 \leq i \leq n_{max}$. Additionally, it contains one domain element x^ϵ for the model fragment I^ϵ.

We first define the interpretation function $\cdot^{\mathcal{I}}$ for concept symbols, based on the concept symbols in the model fragments.

- For every $A \in N_c$, $A^{\mathcal{I}} = \begin{cases} \{x_i^D \mid A \in I^D, i \leq n_{max}\} & \text{if } A \notin I^\epsilon \\ \{x_i^D \mid A \in I^D, i \leq n_{max}\} \cup \{x^\epsilon\} & \text{if } A \in I^\epsilon \end{cases}$

The model fragments I^D serve as guide to assign concepts to the corresponding individuals x_i^D, or to x^ϵ if $D = \epsilon$.

In order to satisfy the transitivity axioms, for each role r we first define an initial interpretation function $r_0^{\mathcal{I}}$ without transitivity, which we then extend using the transitive closure, as we did for $Res_{\mathcal{SH}}$ in Section 5.4:

- For every $r \in N_r$:
  - $r_0^{\mathcal{I}} = \{(x_i^{D_1}, x_j^{D_2}) \mid i \leq n_{max}, j \leq \#\{+1s.D_2 \in I^D, I^{D_2} \neq \emptyset\}\}$
  - $r^{\mathcal{I}} = r_0^{\mathcal{I}} \cup \{(x, x') \in (s_0^{\mathcal{I}})^* \mid s \sqsubseteq_{\mathcal{N}} r, \mathsf{trans}(s) \in \mathcal{N}\}$

Note that roles with a transitive sub-role are not allowed in literals of the form $\leq nr.\neg\mathcal{D}$ with $n > 0$. Therefore, adding the transitive closure does not affect the satisfiability of any number of the form $\leq nr.D$ with $n \geq 1$, and the extension of the candidate model has no different effect than for $Res_{\mathcal{SH}}$.

Lemma 6.2.7. *$\mathcal{I}$ is a model of $\mathcal{N}^*$.*

Proof. The role inclusion and transitivity axioms in $\mathcal{N}^*$ are satisfied by the model for the same reasons as for the candidate models built in Sections 5.4 for $Res_{\mathcal{ALCHF}}$ (see also Lemma 6.2.2).

It remains to show that the remaining clauses are satisfied by the model as well, that is, that $\mathcal{I} \models C$ for all $C \in \mathcal{N}^*$. The proof is again by contradiction. Assume C_i is the smallest clause in $\mathcal{N}^*$ with $\mathcal{I}_{\mathcal{N}^*} \not\models C_i$. We only have to check the cases where the maximal literal in C_i is a number restriction. For the remaining cases the situation is the same as for $Res_{\mathcal{SH}}$.

1. The maximal literal in C_i is of the form $\geq nr.\mathcal{D}$. Since $I^D \models \geq nr.\mathcal{D}$, we have $\#\{+1s.D_j \in I^D, s \sqsubseteq_{\mathcal{N}} r, \{D_j\} \sqsubseteq \mathcal{D}\} \geq n$.

   (a) Suppose $I^{D_j} \neq \emptyset$ for each $D_j \in \{D_j \mid +1s.D_j \in I^D, s \sqsubseteq_{\mathcal{N}} r,\ D_j \sqsubseteq \mathcal{D}\}$. Then there is an r-relation from every x_i^D to at least n domain elements. Note that if $\{D_j\} \sqsubseteq_d \mathcal{D}$, either $D_j \in \mathcal{D}$ or $\neg D_j \sqcup D'_j \in \mathcal{N}^*$ for one $D'_j \in \mathcal{D}$. In both cases, $D'_j \in I^{D_j}$ for some $D'_j \in \mathcal{D}$ and $x_k^{D_j} \in \mathcal{D}^{\mathcal{I}_{\mathcal{N}^*}}$. This implies $x_i^D \in (\geq nr.\mathcal{D})^{\mathcal{I}_{\mathcal{N}^*}}$, which contradicts our initial assumption.

   (b) Suppose $I^{D_2} = \emptyset$ for some $D_2 \in \{D_j \mid +1s.D_j \in I^D, s \sqsubseteq_{\mathcal{N}} r,\ D_j \sqsubseteq \mathcal{D}\}$. Then there is a clause $\neg D_2 \in \mathcal{N}^*$. Additive literals $+1s.D_2 \in I^D$ are only added in Step 4 in the construction of I^D if there is a clause C_j in which the maximal literal is $\geq ms.\mathcal{D}_2$, where D_2 is the maximal definer symbol in $\mathcal{D}_2$, and for which $I^D_{j-1} \not\models C_j$. Applying the $\geq$-resolution rule on $\neg D_2$ and C_j produces the following smaller clause:

$$C_k = (C_j \setminus \{\geq ms.\mathcal{D}_2\}) \cup \{\geq ms.(\mathcal{D}_2 \setminus \{D_2\})\}.$$

   Due to Lemma 6.2.6, $I^D \models C_k$. Since the maximal literal in C_k cannot be maximal in any clause larger than C_k, we obtain $I^D_k \models C_k$. In our ordering, $C_k \prec_c C_j$. But then, $I^D_{j-1} \models C_j$, and no additive literals are added for C_j. We obtain that, contrary to an earlier observation, no additive literal of the form $+1s.D_2$ is added to I^D. We have a contradiction, which means the assumption $I^{D_2} = \emptyset$ cannot be true.

2. The maximal literal in C_i is of the form $\leq nr.\neg D_2$.

   (a) Assume r is simple, that is $\mathsf{trans}(r) \notin \mathcal{N}$ and there is no role $s \sqsubseteq_{\mathcal{N}} r$ with $\mathsf{trans}(s) \in \mathcal{N}$. Then, by Lemmata 6.2.5 and 6.2.6, C_i is satisfied by the model.

   (b) Assume r is not simple, that is $\mathsf{trans}(r) \in \mathcal{N}$ or there is a role s with $s \sqsubseteq_{\mathcal{N}} r$ and $\mathsf{trans}(s) \in \mathcal{N}$. Recall that by the definition of $\mathcal{SHQ}$, only simple roles are allowed in number restrictions of the form $\leq n'r.D'$ with $n' > 0$. Therefore, since r is not simple, $n = 0$, and that the maximal literal in C_i is of the form $\leq 0r.\neg D_2$. In this case, the proof can be done in the same way as for Lemmata 4.2.19 and 5.2.4 for $Res_{\mathcal{ALC}}$ and $Res_{\mathcal{SH}}$.

$\leq$-Combination:

$$\frac{C_1 \sqcup \leq n_1 r_1.\neg\mathcal{D}_1 \qquad C_2 \sqcup \leq n_2 r_2.\neg\mathcal{D}_2 \qquad r \sqsubseteq_{\mathcal{N}} r_1 \qquad r \sqsubseteq_{\mathcal{N}} r_2}{C_1 \sqcup C_2 \sqcup \leq (n_1 + n_2) r.\neg\mathcal{D}_{12}}$$

$\leq\geq$-Combination:

$$\frac{C_1 \sqcup \leq n_1 r_1.\neg\mathcal{D}_1 \qquad C_2 \sqcup \geq n_2 r_2.\mathcal{D}_2 \qquad r_2 \sqsubseteq_{\mathcal{N}} r_1 \qquad n_1 \geq n_2}{\begin{array}{c} C_1 \sqcup C_2 \sqcup \leq (n_1 - n_2) r_1.\neg(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup \geq 1 r_2.\mathcal{D}_{12} \\ \vdots \\ C_1 \sqcup C_2 \sqcup \leq (n_1 - 1) r_1.\neg(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup \geq n_2 r_2.\mathcal{D}_{12} \end{array}}$$

Figure 6.2: Additional inference rules of $Int_{\mathcal{SHQ}}$.

We established that every clause in $\mathcal{N}^*$ is satisfied by the candidate model, which means that $\mathcal{I} \models C$ for every clause $C \in \mathcal{N}^*$, and that $\mathcal{I}$ is a model for $\mathcal{N}^*$. □

Lemma 6.2.7 allows us to establish refutational completeness of $Res_{\mathcal{SHQ}}$. Together with Theorem 6.1.4, we therefore have the following theorem.

Theorem 6.2.8. *$Res_{\mathcal{SHQ}}$ is sound and refutationally complete.*

In a similar fashion as for $Res^s_{\mathcal{ALC}}$ and $Res^s_{\mathcal{ALCH}}$, it can be shown that the redundancy elimination rules in $Res^s_{\mathcal{SHQ}}$ preserve refutational completeness. (Note that the orderings $\prec_l$ and $\prec_d$ used in the model construction are compatible to the subsumption relations $\sqsubseteq_l$ and $\sqsubseteq_d$ defined in Definition 6.1.5.) Termination of $Res^s_{\mathcal{SHQ}}$ is given by Theorem 6.1.6. We can therefore establish that $Res^s_{\mathcal{SHQ}}$ is a sound and refutationally complete decision procedure for satisfiability of $\mathcal{SHQ}\nu$ ontologies.

Theorem 6.2.9. *$Res^s_{\mathcal{SHQ}}$ is sound, refutationally complete and terminating, and provides a decision procedure for $\mathcal{SHQ}\nu$.*

6.3 The Interpolation Calculus

For similar reasons as for $Int_{\mathcal{ALC}}$ (see Section 4.4), additional inference rules are necessary in order to obtain an interpolation complete calculus for $\mathcal{SHQ}$. These rules are shown in Figure 6.2. $Int_{\mathcal{SHQ}}$ extends $Res_{\mathcal{SHQ}}$ by these rules.

Again one can verify that the rules in Figure 6.2 are generalisations of rules we have seen in earlier chapters. The $\leq$-combination rule is a generalisation of the $\forall\forall$-role

propagation rule in $Int^s_{\mathcal{ALCH}}$ (see Sections 5.1.2 and 4.4), and the $\leq\geq$-combination rule is a generalisation of the universalisation rule in $Res_{\mathcal{SIF}}$ (see Section 5.5). The two rules are sufficient to compute the uniform interpolant of any $\mathcal{SHQ}\nu$ ontology.

We first prove soundness of the two new rules.

Lemma 6.3.1. *The $\leq$-combination rule is sound.*

Proof. Assume we have two clauses $C_1 \sqcup \leq n_1 r_1.\neg\mathcal{D}_1, C_2 \sqcup \leq n_2 r_2.\neg\mathcal{D}_2 \in \mathcal{N}$, and we further have a role r with $r_1 \sqsubseteq_\mathcal{N} r$ and $r_2 \sqsubseteq_\mathcal{N} r$. We have to show that every model of $\mathcal{N}$ also satisfies the axiom $\top \sqsubseteq C_1 \sqcup C_2 \sqcup \leq(n_1+n_2)r.\neg(\mathcal{D}_1 \sqcap \mathcal{D}_2)$.

Assume $\mathcal{I}$ is a model of $\mathcal{N}$ and x is a domain element in $\mathcal{I}$ with $x \in (\leq n_1 r_1.\neg\mathcal{D}_1)^\mathcal{I}$ and $x \in (\leq n_2 r_2.\neg\mathcal{D}_2)^\mathcal{I}$. We show that we then also have $x \in (\leq(n_1+n_2)r.\neg(\mathcal{D}_1 \sqcap \mathcal{D}_2))^\mathcal{I}$.

Since $r \sqsubseteq_\mathcal{N} r_1$ and $r \sqsubseteq_\mathcal{N} r_2$, $x \in (\leq n_1 r.\neg\mathcal{D}_1)^\mathcal{I}$ and $x \in (\leq n_2 r.\neg\mathcal{D}_2)^\mathcal{I}$. Denote by X_1 the r-successors of x satisfying $\neg\mathcal{D}_1$ and by X_2 the r-successors of x satisfying $\neg\mathcal{D}_2$. The following equations hold.

$$\#X_1 \leq n_1 \tag{6.8}$$

$$\#X_2 \leq n_2 \tag{6.9}$$

$$\#(X_1 \cup X_2) \leq \#X_1 + \#X_2 \tag{6.10}$$

$$\#(X_1 \cup X_2) \leq n_1 + n_2 \tag{6.11}$$

Each element of $X_1 \cup X_2$ satisfies $\neg\mathcal{D}_1$ or $\neg\mathcal{D}_2$, and therefore satisfies $\neg\mathcal{D}_1 \sqcup \neg\mathcal{D}_2$, which is equivalent to $\neg(\mathcal{D}_1 \sqcap \mathcal{D}_2)$. Hence, x has at most n_1+n_2 r-successors satisfying $\neg(\mathcal{D}_1 \sqcap \mathcal{D}_2)$, and $x \in (\leq(n_1+n_2)r.\neg(\mathcal{D}_1 \sqcap \mathcal{D}_2))^\mathcal{I}$. □

Lemma 6.3.2. *The $\leq\geq$-combination rule is sound.*

Proof. Assume we have two clauses $C_1 \sqcup \leq n_1 r_1.\neg\mathcal{D}_1, C_2 \sqcup \geq n_2 r_2.\mathcal{D}_2 \in \mathcal{N}$ where $n_1 \geq n_2$, and we additionally have $r_2 \sqsubseteq_\mathcal{N} r_1$. We have to prove that every axiom of the following form is entailed by $\mathcal{N}$, where $0 \leq i < n_2$.

$$\top \sqsubseteq C_1 \sqcup C_2 \sqcup \leq(n_1 - n_2 + i)r_1.\neg(\mathcal{D}_1 \sqcup D_2) \sqcup \geq(1+i)r_2.(\mathcal{D}_1 \sqcap \mathcal{D}_2) \tag{6.12}$$

Let $\mathcal{I}$ be any model of $\mathcal{N}$, and let x be a domain element of $\mathcal{I}$ such that $x \in (\leq n_1 r_1.\neg\mathcal{D}_1)^\mathcal{I}$ and $x \in (\geq n_2 r_2.\mathcal{D}_2)^\mathcal{I}$. We have to show that then one of the following is true.

$$x \in (\leq(n_1 - n_2 - i)r_1.\neg(\mathcal{D}_1 \sqcup \mathcal{D}_2))^\mathcal{I} \tag{6.13}$$

$$x \in (\geq(1+i)r_2.(\mathcal{D}_1 \sqcap \mathcal{D}_2))^{\mathcal{I}}. \tag{6.14}$$

Assume (6.14) does not hold. We then have the following.

$$x \in (\leq i r_2.(\mathcal{D}_1 \sqcap \mathcal{D}_2))^{\mathcal{I}}.$$

We show that in this case, (6.13) is satisfied.

Since $r_2 \sqsubseteq_{\mathcal{N}} r_1$ and $x \in (\geq n_2 r_2.\mathcal{D}_2)^{\mathcal{I}}$, we also have $x \in (\geq n_2 r_1.\mathcal{D}_2)^{\mathcal{I}}$. Denote the r_1-successors of x that satisfy $\mathcal{D}_1$ by X_1 and the r_1-successors of x that satisfy $\mathcal{D}_2$ by X_2. Furthermore, denote the set of all r_1-successors of x by X and let $n = \#X$.

Based on the concepts that x satisfies, we can establish the following inequations.

$$\begin{aligned} \#X &= n \\ \#(X \setminus X_1) &\leq n_1 \\ \#X_1 &> n - n_1 \\ \#X_2 &\geq n_2 \\ \#(X_1 \cap X_2) &\leq i \end{aligned}$$

This allows us to infer the maximum cardinality of the set r_1-successors of x that satisfy $\neg(\mathcal{D}_1 \sqcup \mathcal{D}_2)$, which is the set $X \setminus (X_1 \cup X_2)$.

$$\begin{aligned} \#(X_1 \cup X_2) &= \#X_1 + \#X_2 - \#(X_1 \cap X_2) \\ \#(X_1 \cup X_2) &> (n - n_1) + n_2 - i \\ \#(X \setminus (X_1 \cup X_2) &\leq n - ((n - n_1) + n_2 - i) \\ &\leq n_1 - n_2 + i \end{aligned}$$

We established that there are at most $(n_1 - n_2 + i)$ r_1-successors that satisfy $\neg(\mathcal{D}_1 \sqcup \mathcal{D}_2)$, and that 6.13. This proves that the $\leq\geq$-combination rule is sound. □

With Theorem 6.1.4 and Lemmata 6.3.1 and 6.3.2, we can establish the following theorem.

Theorem 6.3.3. *$Int_{\mathcal{SHQ}}$ is sound.*

6.4 Interpolation Completeness of $Int_{\mathcal{SHQ}}$

It can be shown in a similar fashion as for $Res_{\mathcal{ALC}}$ that $Res_{\mathcal{SHQ}}$ is still refutationally complete if we restrict rules to be only applied on maximal literals and to produce clauses with at most one negative definer literal. Denote the resulting calculus by $Res^{norm^{\prec_l}}_{\mathcal{SHQ}}$. However, since $Res_{\mathcal{SHQ}}$ has completely different rules for handling role restrictions, we have to establish a similar lemma for $Int_{\mathcal{SHQ}}$ as we stated for $Int_{\mathcal{ALC}}$ in Lemma 4.4.6. This lemma makes use of the notion of subsuming contexts of definers, which is defined in Definition 4.4.4. The $\mathcal{SHQ}$ normal form allows us to represent equivalent sets of clauses in very different ways that are not captured by our redundancy notion. These syntactical variants of equivalent clauses play a major role in our proof for interpolation completeness. We therefore have to generalise the notion of subsuming contexts.

Definition 6.4.1. A definer D_1 *occurs in* $\mathcal{N}$ *in a deductively subsuming context to a definer* D_2, if there is a set of clauses $\mathcal{N}'$ such that $\mathcal{N} \models \mathcal{N}'$ and D_1 occurs in $\mathcal{N}'$ in a subsuming context to D_2.

Given two clause sets $\mathcal{N}$ and $\mathcal{M}$, a definer D_1 that occurs in $\mathcal{N}$ and a definer D_2 that occurs in $\mathcal{M}$, D_1 occurs in a deductively subsuming context to D_2 if D_1 occurs in a deductively subsuming context in $\mathcal{N} \cup \mathcal{M}$.

Observe that if D_1 occurs in $\mathcal{N}$ in a subsuming context to D_2, then D_1 also occurs in $\mathcal{N}$ in a deductively subsuming context to D_2.

Lemma 6.4.2. *Let* $\mathcal{N}$ *and* $\mathcal{M}$ *be two sets of clauses, and* D_1 *and* D_2 *two definers that are combined in* $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$ *by a definer* D_{12x}*. Then,* D_1 *and* D_2 *are combined by a definer* D'_{12x} *in* $\mathcal{N}^* = Int_{\mathcal{SHQ}}(\mathcal{N})$*, and either* D'_{12x} *occurs in a deductively subsuming context to* D_{12x}*, or* $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N}^* \cup \mathcal{M})$ *contains a clause* $\neg D''_{12x} \sqcup D'_{12x}$*, and* D''_{12x} *occurs in a deductively subsuming context to* D_{12x}*.*

Proof. Let $\mathcal{N}$, $\mathcal{M}$, D_1, D_2 and D_{12x} be as in the lemma, that is, D_1 and D_2 are combined in $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$ by the definer D_{12x}.

Let $C_1, C_2 \in Res^{norm}_{\mathcal{SHQ}}(\mathcal{N})$ be two clauses in which respectively the two definers D_1 and D_2 occur under a role restriction. If a rule of $Res_{\mathcal{SHQ}}$ can be directly applied on C_1 and C_2 to combine D_1 and D_2, D_1 and D_2 are already combined in $Res^{norm}_{\mathcal{SHQ}}(N)$. Suppose therefore this is not the case.

The case in which C_1 and C_2 contain different negative definer literals can be proved using induction, given that we can prove the case where $C_1 \cup C_2$ contains at most one negative definer literal (see proof for Lemma 4.4.6). We may therefore assume that $C_1 \cup C_2$ contains at most one negative definer literal.

D_1 and D_2 are combined in $Res_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$ by D_{12x}. Therefore, there are two clauses $\neg D_{12x} \sqcup D_1$, $\neg D_{12x} \sqcup D_2 \in Res_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$. Let $C_3 \in Res_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$ be a clause that contains the "missing link" for introducing such a definer D_{12x}. That is, C_3 contains a number restriction such that, using the rules of $Res_{\mathcal{SHQ}}$ on C_1, C_2 and C_3, the definer D_{12x} is introduced. We consider the different cases for C_1, C_2 and C_3 and show that in each case, we can introduce the definer D''_{12x} in a deductively subsuming context if we first apply the rules of $Int_{\mathcal{SHQ}}$ on C_1 and C_2, and then saturate the conclusions with C_3 using $Res^{norm}_{\mathcal{SHQ}}$.

1. The maximal literals in C_1 and C_2 are of the form $\leq n_i r_i.\neg D_i$, $i \in \{1, 2\}$, and the maximal clause of C_3 is of the form $\geq n_3 r.D_3$, where $n_3 > n_1 + n_2$, $r \sqsubseteq_{\mathcal{N}} r_1$ and $r \sqsubseteq_{\mathcal{N}} r_2$. By using first the $\geq\leq$-combination rule of $Res^{norm}_{\mathcal{SHQ}}$, we can infer the following clauses.

   1. $C'_1 \sqcup C'_3 \sqcup \geq (n_3 - n_1) r.D_{13}$
   2. $C'_2 \sqcup C'_3 \sqcup \geq (n_3 - n_2) r.D_{23}$
   3. $C'_1 \sqcup C'_2 \sqcup C_3 \sqcup \geq (n_3 - n_1 - n_2) r.D_{123}$.

   D_1 and D_2 are combined in $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$ by D_{123}.

   Using $Int_{\mathcal{SHQ}}$, we can derive the same clause containing $\geq (n_3 - n_1 - n_2) r.D_{123}$ if we apply first the $\leq$-rule on the clauses C_1 and C_2, and then the $\leq\geq$-combination rule on C_3:

   $1'$. $C'_1 \sqcup C'_2 \sqcup \leq (n_1 + n_2) r.\neg D_{12}$
   $2'$. $C'_1 \sqcup C'_2 \sqcup C_3 \sqcup \geq (n_3 - n_1 - n_2) r.D''_{123}$

   D_1 and D_2 are combined in $\mathcal{N}^*$ by D_{12}, $\neg D''_{123} \sqcup D_{12} \in Res^{norm}_{\mathcal{SHQ}}$, and D''_{123} occurs in a deductively subsuming context to D_{123}.

2. The maximal literals in C_1 and C_3 are of the form $\geq n_i r_i.D_i$, $i \in \{1, 3\}$ and the maximal literal in C_2 is of the form $\leq n_2 r.\neg D_2$, where $n_i \leq n_2 \leq n_1 + n_3$ and

$r_i \sqsubseteq r$ for $i \in \{1, 3\}$. D_1 and D_2 are not combined in $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N})$, since $n_1 < n_2$, and the $\geq\leq$-combination rule is not applicable for this case.

The following clauses are derived by applying the $\geq$-combination rule on C_1 and C_3:

$$C'_1 \sqcup C'_3 \sqcup \geq(n_1 + n_3)r.(D_1 \sqcup D_3) \sqcup \geq 1r_i.D_{13}$$
$$\vdots$$
$$C'_1 \sqcup C'_3 \sqcup \geq n'r.(D_1 \sqcup D_3) \sqcup \geq n''r_i.D_{13},$$

where $i \in \{1, 3\}$, $n' = max(n_1, n_3) + 1$ and $n'' = min(n_1, n_3)$. Observe that $n_2 \geq n'$, since n_2 is the largest of the three initial numbers n_1, n_2 and n_3. The $\geq\leq$-combination rule can be applied on each of these clauses and C_3. This way, we obtain the following set of clauses:

$$\begin{aligned} C'_1 \sqcup C'_2 \sqcup C'_3 \sqcup \geq(n_1 + n_3 - n_2)r.(D_{12} \sqcup D_{23}) \sqcup \geq 1r_i.D_{13} \\ \vdots \\ C'_1 \sqcup C'_2 \sqcup C'_3 \sqcup \geq 1r.(D_{12} \sqcup D_{23}) \sqcup \geq(n_1 + n_3 - n_2)r_i.D_{13} \end{aligned} \tag{6.15}$$

D_1 and D_2 are combined by D_{12}, which represents $D_1 \sqcap D_2$.

The last two literals in these clauses describe all possible distributions of $n = n_1 + n_3 - n_2$ r-successors to sets of individuals satisfying $D_{12} \sqcup D_{23}$ and individuals satisfying D_{13}. If a domain element satisfies both $D_{12} \sqcup D_{23}$ and D_{13}, it satisfies all three definer concepts D_{12}, D_{13} and D_{23}. Assume we abbreviate $D_{12} \sqcap D_{13} \sqcap D_{23}$ by D_{123}. Clause Set (6.15) is then equivalent to the following set of clauses:

$$\begin{aligned} C'_1 \sqcup C'_2 \sqcup C'_3 \sqcup \geq(n_1 + n_3 - n_2)r.(D_{12} \sqcup D_{23} \sqcup D_{13}) \sqcup \geq 1r_i.D_{123} \\ \vdots \\ C'_1 \sqcup C'_2 \sqcup C'_3 \sqcup \geq 1r.(D_{12} \sqcup D_{23} \sqcup D_{13}) \sqcup (n_1 + n_3 - n_2)r_i.D_{123} \end{aligned} \tag{6.16}$$

Note that here D_{123} is not an introduced definer, but a place-holder for the concept $D_{12} \sqcap D_{13} \sqcap D_{23}$.

We show how a stronger set of clauses can be inferred by using first $Int_{\mathcal{SHQ}}$ on C_1 and C_2. First, the $\leq\geq$-combination rule is applied on C_1 and C_2, which produces

the following clauses:

$$C'_1 \sqcup C'_2 \sqcup \leq (n_2 - n_1) r_3.\neg(D_1 \sqcup D_2) \sqcup \geq 1 r_1.D'_{12}$$
$$\vdots$$
$$C'_1 \sqcup C'_2 \sqcup \leq (n_2 - 1) r_3.\neg(D_1 \sqcup D_2) \sqcup \geq n_1 r_1.D'_{12}.$$

D_1 and D_2 are combined in $\mathcal{N}^*$ by D'_{12}. D'_{12} represents the same conjunction as D_{12}, namely $D_1 \sqcap D_2$. We show that D'_{12} occurs in $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N}^* \cup \mathcal{M})$ in an deductively subsuming context to D_{12} in $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$.

To make the following easier, we assume that we have in addition the following redundant clauses:

$$C'_1 \sqcup C'_2 \sqcup \leq n_2 r_3.\neg(D_1 \sqcup D_2) \sqcup \geq (n_1 + 1) r_1.D'_{12}$$
$$\vdots$$
$$C'_1 \sqcup C'_2 \sqcup \leq (n_3 - 1) r_3.\neg(D_1 \sqcup D_2) \sqcup \geq (n_3 - (n_2 - n_1)) r_1.D'_{12}.$$

Since $n_3 > n_2$ and $r_3 \sqsubseteq r$, these are all subsumed by $C_2 \sqcup \leq n_2 r.\neg D_2$ and therefore redundant. Therefore, adding these clauses does not lead to any new inferences that are not entailed by the original set of clauses.

By applying the $\geq\leq$-combination rule on C_3 and each of these clauses, we obtain the following clauses.

$$C'_1 \sqcup C'_2 \sqcup C'_3 \sqcup \geq (n_3 - (n_2 - n_1)) r_3.(D_{13} \sqcup D_{23}) \sqcup \geq 1 r_1.D'_{12} \tag{6.17}$$
$$\vdots \tag{6.18}$$
$$C'_1 \sqcup C'_2 \sqcup C'_3 \sqcup \geq 1 r_3.(D_{13} \sqcup D_{23}) \sqcup \geq (n_3 - (n_2 - n_1)) r_1.D'_{12}. \tag{6.19}$$

Again the last two literals in these clauses describe all possible distributions of $n = n_3 - (n_2 - n_1) = n_1 + n_3 - n_2$ r-successors to sets of individuals satisfying D_{12}, D_{13} and D_{23}. Observe that the individuals satisfying D_{12} are all r_1-successors, and the individuals satisfying $(D_{13} \sqcap D_{23})$ are all r_3-successors. Therefore, the intersection of both sets always consists of elements that are both r_1-successors and r_3-successors. Hence, if we replace D'_{12} in Clause Set (6.17) by D_{12}, we obtain a set of clauses that the logically entails Clause Set (6.16). Therefore, D'_{12} occurs in a deductively subsuming context in $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N}^* \cup \mathcal{M})$ to D_{12}.

In the remaining cases, either D_1 and D_2 are already combined in $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N})$, or they are also not combined in $Res^{norm}_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$. □

Theorem 6.4.3. *The calculus $Int_{\mathcal{SHQ}}$ is interpolation complete for forgetting concept symbols in $\mathcal{SHQ}$.*

Proof. The proof goes analogously to the proof for interpolation completeness of $Int_{\mathcal{ALC}}$ (Theorem 4.4.11). If $Int_{\mathcal{SHQ}}$ is interpolation complete for forgetting concept symbols in $\mathcal{SHQ}$, then for any $\mathcal{SHQ}$ ontology $\mathcal{O}$ and signature $\mathcal{S}$ with $N_r \subseteq \mathcal{S}$, $\mathcal{O}^{\mathcal{S}}\,Ont(\mathcal{N}^{\mathcal{S}})$ is an $\mathcal{SHQ}$ uniform interpolant of $\mathcal{O}$ for $\mathcal{S}$, where $\mathcal{N}^{\mathcal{S}} = (Int_{\mathcal{ALC}}(Cl(\mathcal{O})))_{\mathcal{S}}$. This means, for every $\mathcal{SHQ}$ axiom $C \sqsubseteq D$ with $sig(C \sqsubseteq D) \subseteq \mathcal{S}$, we should have $\mathcal{O} \models C \sqsubseteq D$ iff $\mathcal{O}^{\mathcal{S}} \models C \sqsubseteq D$. Since $Int_{\mathcal{SHQ}}$ is sound, $\mathcal{O}^{\mathcal{S}} \models C \sqsubseteq D$ implies $\mathcal{O} \models C \sqsubseteq D$. For the other direction, since $Res^{norm\prec_l}_{\mathcal{SHQ}}$ is refutationally complete, we should be able to derive the empty clause from $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$ using $Res^{norm\prec_l}_{\mathcal{SHQ}}$, where $\mathcal{M} = Cl(\exists r^*.(C \sqcap \neg D))$ and $r^* \notin sig(\mathcal{O})$, if we can do it from $\mathcal{N} \cup \mathcal{M}$, where $\mathcal{N} = Cl(\mathcal{O})$. Choose for $\prec_l$ an ordering that prefers symbols outside of $\mathcal{S}$. Due to Lemma 6.4.2, all definers in $Res^{norm\prec_l}_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$ with $\neg D \in Res^{norm\prec_l}_{\mathcal{SHQ}}(\mathcal{N} \cup \mathcal{M})$ are introduced in a deductively subsuming contexts when $Res^{norm\prec_l}_{\mathcal{SHQ}}(\mathcal{N}^{\mathcal{S}} \cup \mathcal{M})$ is computed. All remaining inferences that are on symbols outside of $\mathcal{S}$ are performed first by $Res^{norm\prec_l}_{\mathcal{SHQ}}$ and are already computed in $\mathcal{N}^{\mathcal{S}}$. Hence, if the empty clause can be derived from $\mathcal{N} \cup \mathcal{M}$ using $Res^{norm\prec_l}_{\mathcal{SHQ}}$, then it can also be derived from $\mathcal{N}^{\mathcal{S}} \cup \mathcal{M}$. □

6.5 Examples

Example 6.5.1 (The Bicycle Example)**.** Assume we have an ontology $\mathcal{O}_{bike}$ of the following form:

$$\mathsf{Bicycle} \sqsubseteq \exists \mathsf{hasWheel}.\mathsf{FrontWheel} \sqcap \exists \mathsf{hasWheel}.\mathsf{RearWheel}$$
$$\mathsf{FrontWheel} \sqsubseteq \mathsf{Wheel} \sqcap \neg \mathsf{RearWheel}$$
$$\mathsf{RearWheel} \sqsubseteq \mathsf{Wheel} \sqcap \neg \mathsf{FrontWheel}$$

Since FrontWheel and RearWheel are disjoint, the ontology entails that Bicycle has at least two wheels. In order to validate that our method preserves this information, we compute a uniform interpolant for $\mathcal{S} = \{\mathsf{Bicycle}, \mathsf{Wheel}, \mathsf{hasWheel}\}$. The $\mathcal{SHQ}$ normal

form $\mathcal{N}_{bike}$ of this ontology is the following:

1. $\neg\mathsf{Bicycle} \sqcup \geq 1\mathsf{hasWheel}.D_1$

2. $\neg D_1 \sqcup \mathsf{FrontWheel}$

3. $\neg\mathsf{Bicycle} \sqcup \geq 1\mathsf{hasWheel}.D_2$

4. $\neg D_2 \sqcup \mathsf{RearWheel}$

5. $\neg\mathsf{FrontWheel} \sqcup \mathsf{Wheel}$

6. $\neg\mathsf{FrontWheel} \sqcup \neg\mathsf{RearWheel}$

7. $\neg\mathsf{RearWheel} \sqcup \mathsf{Wheel}$

We first apply resolution on FrontWheel.

8. $\neg D_1 \sqcup \mathsf{Wheel}$ *(Resolution 2, 5)*

9. $\neg D_1 \sqcup \neg\mathsf{RearWheel}$ *(Resolution 2, 6)*

The combination rules do not make further inferences on FrontWheel possible. We can therefore directly proceed to compute inferences on RearWheel. From the current clause set, only one resolvent with less than two negative definer literals can be inferred.

10. $\neg D_2 \sqcup \mathsf{Wheel}$ *(Resolution 4, 7)*

By applying the $\geq$-combination rule, additional resolution steps on RearWheel become possible.

~~11.~~ $\neg\mathsf{Bicycle} \sqcup \geq 2\mathsf{hasWheel}(D_1 \sqcup D_2)$
$\sqcup \geq 1\mathsf{hasWheel}.D_{12}$ *($\geq$-Combination 1, 3)*

~~12.~~ $\neg D_{12} \sqcup D_1$

~~13.~~ $\neg D_{12} \sqcup D_2$

~~14.~~ $\neg D_{12} \sqcup \mathsf{Wheel}$ *(Resolution 12, 8)*

~~15.~~ $\neg D_{12} \sqcup \neg\mathsf{RearWheel}$ *(Resolution 12, 9)*

~~16.~~ $\neg D_{12} \sqcup \mathsf{RearWheel}$ *(Resolution 13, 4)*

17. $\neg D_{12}$ *(Resolution 15, 16)*

Clause 17 subsumes the Clause 12–16. Clause 11 becomes redundant if the $\geq$-resolution is applied with the newly derived unit clause $\neg D_{12}$.

~~18.~~ $\neg\mathsf{Bicycle} \sqcup \geq 2\mathsf{hasWheel}(D_1 \sqcup D_2)$
$\sqcup \geq 1\mathsf{hasWheel}.\bot$ *($\geq$-Resolution 11, 17)*

19. $\neg\mathsf{Bicycle} \sqcup \geq 2\mathsf{hasWheel}(D_1 \sqcup D_2)$ *($\geq$-Elimination on 18)*

Clause 18 subsumes Clause 11, and Clause 19 subsumes Clause 18. After removing all redundant clauses, and clauses that are of the form $\neg D_a \sqcup D_b$ or contain the concept symbol RearWheel, we obtain the set $\mathcal{N}^{\mathcal{S}}$ that contains the Clauses 1, 3, 8, 10, 17 and 19 (the clauses with a bold face number). Eliminating all definers results in the following ontology.

$$\top \sqsubseteq \neg\mathsf{Bicycle} \sqcup \geq 2\mathsf{hasWheel}.(\mathsf{Wheel} \sqcup \mathsf{Wheel})$$
$$\top \sqsubseteq \neg\mathsf{Bicycle} \sqcup \geq 1\mathsf{hasWheel}.\mathsf{Wheel}$$

The last axiom is redundant, and the first one can be reformulated to the following axiom.

$$\mathsf{Bicycle} \sqsubseteq \geq 2\mathsf{hasWheel}.\mathsf{Wheel}$$

This is the uniform interpolant $\mathcal{T}_{bike}^{\mathcal{S}}$ of $\mathcal{T}_{bike}$ for $\mathcal{S} = \{\mathsf{Bicycle}, \mathsf{hasWheel}, \mathsf{Wheel}\}$, as we observed at the beginning of the chapter.

Example 6.5.2 (Example with $\leq$-Restrictions)**.** Suppose we have the following ontology $\mathcal{O}_{\mathcal{SHQ}}$.

$$A_1 \sqsubseteq \geq 5r.(A \sqcup B)$$
$$A_2 \sqsubseteq \leq 3r.A$$

The $\mathcal{SHQ}$ normal form $\mathcal{N}_1$ of $\mathcal{T}_1$ consists of the following clauses:

1. $\neg A_1 \sqcup \geq 5r.D_1$

2. $\neg D_1 \sqcup A \sqcup B$

3. $\neg A_2 \sqcup \leq 3r.\neg D_2$

4. $\neg D_2 \sqcup \neg A$.

Suppose we want to eliminate A. We first apply the $\geq\leq$-combination rule on Clauses 1 and 3.

5. $\neg A_1 \sqcup \neg A_2 \sqcup \geq 2r.D_{12}$	*($\geq\leq$-Combination 1, 3)*
6. $\neg D_{12} \sqcup D_1$	*($D_{12} \sqsubseteq D_1$)*
7. $\neg D_{12} \sqcup D_2$	*($D_{12} \sqsubseteq D_2$)*
~~8.~~ $\neg D_{12} \sqcup A \sqcup B$	*(Resolution 2, 6)*
9. $\neg D_{12} \sqcup \neg A$	*(Resolution 7, 4)*
10. $\neg D_{12} \sqcup B$	*(Resolution 8, 9)*

$\mathcal{N}_1^{-A}$ consists of the clauses with a bold face number. Eliminating all definers results in the following set of axioms.

$$\bot \sqsubseteq \neg A_1 \sqcup \geq 5r.\top \qquad \bot \sqsubseteq \neg A_2 \sqcup \leq 3r.\neg\top$$
$$\bot \sqsubseteq \neg A_1 \sqcup \neg A_2 \sqcup \geq 2r.B$$

The second axiom is tautological, since $\leq 3r.\neg\top$ is always satisfied. The uniform interpolant of $\mathcal{O}_{\mathcal{SHQ}}^{-A}$ can therefore be represented in the following way.

$$A_1 \sqsubseteq \geq 5r.\top \qquad A_1 \sqcap A_2 \sqsubseteq \geq 2r.B$$

Chapter 7

Uniform Interpolation with ABoxes

Whereas the previous sections only deal with ontologies composed of a TBox and an RBox, in this chapter we present a method for knowledge bases that additionally have an ABox. We focus on $\mathcal{SHI}$ knowledge bases without equality axioms for individuals. In order to reason with both TBox and ABox axioms, the calculus uses a simple form of unification. Examples indicate that extending the underlying description logic with greatest fixpoint operators is not sufficient in order to express the uniform interpolants of knowledge bases finitely. This situation is not unique to uniform interpolation, but has been found in other areas dealing with the adaptation of knowledge bases. In Liu et al. (2011), it is shown that when applying updates or knowledge revision on knowledge bases in a description logic between $\mathcal{ALC}$ or $\mathcal{ALCQI}$, the resulting knowledge base might have to be expressed in a description logic that supports either nominals, Boolean ABox assertions or the '@'-operator from hybrid logics. Similarly, we offer to ways of representing uniform interpolants of $\mathcal{SHI}$ knowledge bases. One way is to use disjunctive ABox assertions, the other is to use nominals.

Using nominals might be preferable over using disjunctive ABox assertions in applications that require the uniform interpolant to be expressed in a common language that is compatible with modern description logic reasoners and the OWL standard. For this reason, our method for computing uniform interpolants of ontologies with ABoxes works in two stages. In the first stage, a uniform interpolant with disjunctive ABox assertions in computed. In the second stage, we approximate this uniform interpolant into a knowledge base with classical ABox and nominals. This approximation

preserves all entailments in the desired signature that can be expressed without disjunctive ABox assertions, and therefore fulfils all conditions of a uniform interpolant.

There are various reasons that make uniform interpolation of knowledge bases worth investigating as part of the thesis. An important application of uniform interpolation with knowledge bases is in privacy and information hiding. This application, discussed in Section 1.1, is motivated by ontology supported systems that store sensitive information. Uniform interpolation may help in eliminating this information without affecting entailments that are not confidential. In a lot of ontology-supported systems, confidential information is stored in ABoxes. Supporting knowledge bases with ABoxes therefore broadens the applicability of uniform interpolation for this particular application.

Second, observe that most applications for uniform interpolation on ontologies also apply to knowledge bases. This is especially true since replacing an ontology in a knowledge base with its uniform interpolant does not necessarily preserve all entailments in the desired signature, even if the ABox itself is in that signature. Consider for example the following ontology $\mathcal{O}_7$:

$$A \sqsubseteq \forall r.B$$
$$A \sqsubseteq \forall s.(\neg B \sqcup C)$$

For $\mathcal{S}_7 = \{A, r, s, C\}$, the $\mathcal{ALC}$ uniform interpolant $\mathcal{O}_7^{\mathcal{S}_7}$ of $\mathcal{O}_7$ is simply an empty set of axioms, since r and s are not connected to each other in any way, and there are no non-tautological $\mathcal{ALC}$ entailments of $\mathcal{O}_7$ that do not use the concept symbol B. However, suppose $\mathcal{O}_7$ is occurs in a knowledge base together with the following ABox $\mathcal{A}_7$:

$$A(a)$$
$$r(a, b)$$
$$s(a, b)$$

$(\mathcal{T}_7, \mathcal{A}_7) \models C(b)$, but $(\mathcal{T}_7^{\mathcal{S}_7}, \mathcal{A}_7) \not\models C(b)$, even though $C \in \mathcal{S}_7$. The reason is that $\mathcal{O}_7$ also entails the axiom $A \sqsubseteq \forall(r \sqcap s).C$. This kind of axiom cannot be expressed in any description logic considered in the thesis. It is also not possible to represent it in $\mathcal{SHOIN}(D)$ or $\mathcal{SROIQ}(D)$, the description logics underlying OWL DL and

OWL 2.0, and therefore does not have to be entailed by any uniform interpolant in these languages.

For this reason, if uniform interpolation is used to compute logical differences, for ontology analysis, or for other applications, it is worth interpolating the whole knowledge base and not only the ontology. For this, a method is required that can compute uniform interpolants of knowledge bases with ABoxes.

In the context of the thesis, we focus on uniform interpolation for ABoxes in languages without equality. This means ABox assertions of the form $a = b$ or $a \neq b$ are not considered as part of our language, and we do not consider description logics with functional role restrictions or number restrictions, which can be used to indirectly express equalities and inequalities between individuals. Instead, we present a method for $\mathcal{SHI}$ knowledge bases without equality assertions. Uniform interpolation for more expressive languages is future work. (Note that, unless these axioms occur in the input, excluding assertions of the form $a = b$ or $a \neq b$ is not a major limitation in practice, since they can be computed using existing classification procedures.)

There are some technical aspects unique to uniform interpolation of knowledge bases. First of all, our calculi have to be extended to perform inferences from ABox assertions. This could be done by adding new rules for every combination of TBox and ABox premises. But since $Int_{\mathcal{SH}}$ and $Int_{\mathcal{ALCHI}}$ have together already nine inference rules only for TBox axioms, this would result in a complex calculus. A more concise representation of the calculus is obtained by using a simple form of unification, which allows us to define generalised rules that can be applied to either ABox or TBox premises or combinations of these.

Second, for the description logics considered in this thesis, axioms in the TBox can only describe tree-shaped structures. This is also reflected in the structure of TBox axioms, whose syntactical form can be represented as a tree with role restrictions being edges, and by the fact that for every TBox we can always build a model of tree-like form, even in the presence of inverse roles (see for example the model construction used in Section 5.6). These models are not real trees, since there are inverse roles and transitive roles, but they have a tree-like shape. On the other hand, with ABoxes, explicit role assertions allow us to construct a structure that is an arbitrary graph and not necessarily in a tree-like shape.

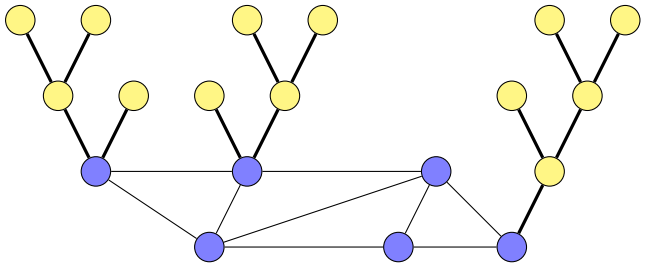

Figure 7.1: Graph structure of an exemplary ABox interpretation.

Fortunately, though not tree-like, models of $\mathcal{SHI}$ always have a forest-like shape. The necessarily "non tree-shaped" part in models of knowledge bases is finite and bounded by the edges and nodes explicitly defined in the ABox, and all additional edges and nodes can still be constructed in a tree-like fashion, if we consider the individuals of the ABox as roots (see Figure 7.1). It is for this reason that we can adapt our model construction approaches used before, in order to prove completeness of the calculus developed in this section.

A more crucial effect of not having the tree-like-model property is that we may have to preserve entailments in the uniform interpolant that do not have a tree-like shape. This is illustrated by the following knowledge base $\mathcal{K}_8$ and the signature $\mathcal{S}_8 = \{A, r, s\}$:

$$\top \sqsubseteq A \sqcup \forall r.(A \sqcup B)$$
$$r(a, b)$$
$$\neg B(b)$$
$$s(a, b)$$

We can infer from this knowledge base that either a or b satisfies A. It is not clear how to express this information using classical ABox assertions in $\mathcal{SHI}$, since the only axiom type that can refer to more than one individual is the role assertion. According to the definition of $\mathcal{SHI}$ uniform interpolants (Section 3.2), we only have to preserve entailments that can be expressed in a single axiom. If the last axiom of the knowledge base, $s(a, b)$, were not part of the knowledge base, we could capture all entailed concept assertions in the desired signature using the two concept assertions $(A \sqcup \exists r.A)(a)$ and $(A \sqcup \exists r^{-}.A)(b)$.

However, the additional role assertion $s(a, b)$ creates a problem. Because of the two role assertions in $\mathcal{K}$, its models do not have a tree-like shape (see Figure 7.2).

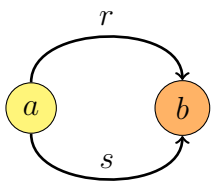

Figure 7.2: Model of a simple non-tree shaped ABox.

Even if we consider only entailments of the form $C(a)$ and $C(b)$, the fact that a has both an r-edge and s-edge leading to the same individual has an impact on the $\mathcal{SHI}$ entailments of the knowledge base. For instance, there is an infinite set of entailments of the following form, where C is any concept in the signature $\mathcal{S}_8$:

$$(A \sqcup \exists r.(A \sqcap C) \sqcup \exists s.(A \sqcap \neg C))(a)$$

For any concept expression C, b either satisfies C or $\neg C$. From this follows that $(\exists r.C \sqcup \exists s.\neg C)(a)$ is true in any model of the ABox. Moreover, if a does not satisfy A, b is both an r-successor and an s-successor of a that satisfies A, and we can identify these successors as satisfying either $(A \sqcap C)$ or $(A \sqcap \neg C)$. In $\mathcal{SHI}\nu$, there is no finite set of TBox axioms and classical ABox assertions in $\mathcal{S}$ that entails all these axioms, since in a single ABox assertion or TBox axiom in $\mathcal{SHI}$, we can only represent tree-like structures. However, there are two possible extensions to the language that make a finite representation of the uniform interpolant possible.

The first extension is Boolean ABox assertions, which have been investigated in Areces et al. (2003). With Boolean ABox assertions, arbitrary concept assertions can be combined using the standard Boolean connectives $\neg$, $\vee$ and $\wedge$. In our particular example, this allows us to represent the uniform interpolant using the following three ABox assertions:

$$A(a) \vee A(b)$$
$$r(a, b)$$
$$s(a, b)$$

However, a drawback of Boolean ABox assertions is that they are not directly supported by standard description logic reasoners, and also not by the web ontology language standard OWL. As shown in Areces et al. (2003), Boolean ABox assertions

can always be represented as classical concept assertions using additional role symbols. While this may be a solution for some applications, a disadvantage is that the resulting knowledge base uses a lot of additional symbols. Also, differently to fixpoint expressions, it is not straightforward how to obtain an approximation of the uniform interpolant that is both completely in the desired signature and that can be processed by a standard reasoner. Applications of uniform interpolation such as computing logical difference (see Section 1.1), require the computed ontology both to be completely in the desired signature and to be processable by a reasoner.

An alternative to using Boolean ABox assertions is to use nominals in the result. In the example, since a and b are connected by an r-edge, we can express the fact that either a or b satisfy A using the concept assertion $(A \sqcup \exists r.(A \sqcap \{b\}))(a)$. If a and b are not connected by a role, we do not have to preserve the information that either a or b satisfy A, since it does not contribute to any additional entailments that can be expressed as classical $\mathcal{SHI}$ entailment. However, in more complex ontologies than in our example, it is necessary to perform additional computations to ensure that this is the case.

Our method for computing uniform interpolants of $\mathcal{SHI}$ knowledge bases proceeds in two stages. In the first stage, we compute the uniform interpolant in a language that allows for Boolean ABox assertions. This stage is described in Section 7.1. In the second stage, these uniform interpolants are approximated into $\mathcal{SHOI}\nu$ knowledge bases that preserve all classical $\mathcal{SHI}$ entailments in the respective signature. This stage is described in Section 7.3.

7.1 The Calculus

In order to represent uniform interpolants of knowledge bases, a restricted form of Boolean ABoxes is sufficient, which we refer to as disjunctive ABoxes.

Definition 7.1.1. A *disjunctive $\mathcal{L}$ concept assertion* is an expression of the form $C_1(a_1) \vee \ldots \vee C_n(a_n)$, where $a_1, \ldots, a_n \in N_i$ and $C_1, \ldots, C_n$ are any $\mathcal{L}$ concepts. A disjunctive concept assertion $C_1(a_1) \vee \ldots \vee C_n(a_n)$ is true in an interpretation $\mathcal{I}$ if $\mathcal{I} \models C_i(a_i)$ for some $1 \leq i \leq n$. A *disjunctive $\mathcal{L}$ ABox* is an $\mathcal{L}$ ABox that additionally contains disjunctive $\mathcal{L}$ concept assertions. For any description logic $\mathcal{L}$, its extension

that allows for disjunctive ABoxes is referred to by $\mathcal{L}_\vee$.

The calculus we introduce in this section can be used to reason on $\mathcal{SHI}\nu_\vee$ knowledge bases. As for the calculi presented in earlier chapters, it only works on normalised inputs. In order to make a more concise representation of the rules possible, we use a representation using variables and disjunctions for normalised knowledge bases.

Definition 7.1.2. A $\mathcal{SHI}$ *concept literal* is a concept of the form A, $\neg A$, $\exists R.D$, $\forall R.D$, where $A \in N_c$, $D \in N_d$ and R is of the form r or r^-, $r \in N_r$. A knowledge base $\mathcal{K}$ is in $\mathcal{SHI}_\vee$ *normal form* if every axiom in $\mathcal{K}$ is a role assertion, a $\mathcal{SHI}$ RBox axiom, or a *clause* of one of these two forms, where L_i is a concept literal and $a_i \in N_i$ for $1 \leq i \leq n$.

1. *TBox clause:* $L_1(x) \vee \ldots \vee L_n(x)$
2. *ABox clause:* $L_1(a_1) \vee \ldots \vee L_n(a_n)$

We additionally require that for any role assertion $R(a, b) \in \mathcal{K}$ there is a corresponding role assertion $\mathsf{Inv}(R)(b, a) \in \mathcal{K}$, and that for any transitiviy axiom $\mathsf{trans}(R) \in \mathcal{K}$ there is a corresponding transitivity axiom $\mathsf{trans}(\mathsf{Inv}(R))$. We call the disjuncts of TBox and ABox clauses respectively *TBox literals* and *ABox literals*. A literal of the form $\neg D(t)$, $t \in N_i \cup \{x\}$, is called negative definer literal. A TBox clause is *normal* if it contains at most one negative definer literal. An ABox clause is normal if it contains no negative definer literal. It is assumed that clauses are represented respectively as sets of TBox and ABox literals, that is, they do not have duplicate literals and their order is not important.

A TBox clause $L_1(x) \vee \ldots \vee L_n(x)$ represents the equivalent concept inclusion $\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n$. The symbol x occurring in TBox clauses is referred to as *variable*. Observe that in contrast to first-order clausal forms, one variable x is sufficient in this representation. Elements of the set $N_i \cup \{x\}$ are called *terms*.

$\mathcal{SHI}\nu_\vee$ knowledge bases can be normalised into $\mathcal{SHI}_\vee$ normal form using adaptions of the rules introduced in Section 4.1. If in a set of clauses every clause is normal, we can eliminate all definers using adaptions of the transformations introduced in Section 4.1.2, or approximate it into a $\mathcal{SHI}_\vee$ knowledge base using the techniques introduced in Section 4.1.3.

Example 7.1.3. Consider the following knowledge base $\mathcal{K}_9$:

$$A \sqsubseteq \forall r.(B \sqcap C)$$

$$r(a, b)$$

$$s(a, b)$$

$$\neg(A \sqcap B)(b)$$

The $\mathcal{SHI}_\vee$ normal form of $\mathcal{K}_9$ is the following set $\mathcal{N}_9$ of clauses.

1. $\neg A(x) \vee (\forall r.D_1)(x)$
2. $\neg D_1(x) \vee B(x)$
3. $\neg D_1(x) \vee C(x)$
4. $r(a, b)$
5. $s(a, b)$
6. $\neg A(b) \vee \neg B(b)$

Our normal form allows for a simple form of unification. In our setting, given two terms t_1 and t_2, the *unifier* of t_1 and t_2 is a substitution that replaces t_1 by t_2, or vice versa. Two terms t_1 and t_2 only have a unifier if $t_1 = x$, $t_2 = x$ or $t_1 = t_2$. For example, the terms a and b do not have a unifier, and the unifier of a and x is $\sigma = [x/a]$. Applied to the clause $C = A(x) \vee B(x)$, this unifier produces the clause $C\sigma = A(a) \vee B(a)$.

The inference rules of the calculus $Int_{\mathcal{SHI}_\vee}$ are shown in Figure 7.3. The refutation calculus $Res_{\mathcal{SHI}_\vee}$ contains all inference rules in Figure 7.3 except for the $\forall\forall$-role propagation rule, the role hierarchy rule and the monotonicity rules. Except for the last two rules, the inference rules are all generalisations and adaptions of the inference rules of $Int_{\mathcal{SH}}$ and $Int_{\mathcal{SIF}}$, excluding those that involve functional role restrictions. Whereas the rules of these calculi can only be applied on TBox clauses and RBox axioms, the inference rules of $Res_{\mathcal{SHI}_\vee}$ and $Int_{\mathcal{SHI}_\vee}$ use unification, and can be applied on premises of which one or both are ABox clauses. In addition, the calculus has rules to handle role assertions. The role instantiation rule is similarly motivated as the $\forall\exists$-role propagation rule, and propagates information in universal role restrictions $\forall R.D$ along role assertions. Observe that, for a role assertion $R(a, b)$, the rule is only applicable to clauses of the forms $C \vee (\forall R.D)(x)$ and $C \vee (\forall R.D)(a)$. The role assertion monotonicity rule follows the same idea as the other monotonicity rules.

Resolution

$$\frac{C_1 \vee A(t_1) \qquad C_2 \vee \neg A(t_2)}{(C_1 \vee C_2)\sigma}$$

$\forall\exists$-Role Propagation

$$\frac{C_1 \vee (\forall R.D_1)(t_1) \qquad C_2 \vee (\exists S.D_2)(t_2) \qquad S \sqsubseteq_{\mathcal{N}} R}{(C_1 \vee C_2)\sigma \vee \exists S.D_{12}(t_1\sigma)}$$

$\forall\forall$-Role Propagation

$$\frac{C_1 \vee (\forall R_1.D_1)(t_1) \qquad C_2 \vee (\forall R_2.D_2)(t_2) \qquad R \sqsubseteq_{\mathcal{N}} R_1 \qquad R \sqsubseteq_{\mathcal{N}} R_2}{(C_1 \vee C_2)\sigma \vee \forall R.D_{12}(t_1\sigma)}$$

$\exists$-Elimination

$$\frac{C \vee (\exists R.D)(t) \qquad \neg D(x)}{C}$$

$\exists$-Monotonicity

$$\frac{C \vee (\exists R.D)(t) \qquad R \sqsubseteq S}{C \vee (\exists S.D)(t)}$$

$\forall$-Monotonicity

$$\frac{C \vee (\forall R.D)(t) \qquad S \sqsubseteq R}{C \vee (\forall S.D)(t)}$$

Role Hierarchy

$$\frac{R_1 \sqsubseteq R_2 \qquad R_2 \sqsubseteq R_3}{R_1 \sqsubseteq R_3}$$

Transitivity

$$\frac{C \vee (\forall R.D)(t) \qquad \mathsf{trans}(S) \qquad for S \sqsubseteq_{\mathcal{N}} R}{C \vee (\forall S.D')(t) \qquad \neg D'(x) \vee D(x) \qquad \neg D'(x) \vee (\forall S.D')}$$

Role Inversion

$$\frac{D_1(x) \vee (\forall R.D_2)(x)}{D_2(x) \vee (\forall \mathsf{Inv}(R).D_1)(x)}$$

Role Instantiation

$$\frac{C_1 \vee (\forall R.D)(t_1) \qquad R(t_2, b)}{C_1\sigma \vee D(b)}$$

Role Assertion Monotonicity

$$\frac{R(a,b) \qquad R \sqsubseteq S}{S(a,b)}$$

where σ is the unifier of t_1 and t_2 if it exists, D_{12} is a possibly new definer representing $D_1 \sqcap D_2$.

Figure 7.3: Inference rules of the calculi $Res_{\mathcal{SHI}_\vee}$ and $Int_{\mathcal{SHI}_\vee}$.

As with $Int_{\mathcal{SIF}}$, it is necessary to apply rules in several stages (see Section 5.5). In the first stage, the initial clause set $\mathcal{N}$ is saturated using the transitivity rule. After this stage, we apply the following transformation on TBox clauses containing a universal role restriction, where one transformation is required for every universal role restriction in the clause:

$$C \vee (\forall R.D)(x) \Longrightarrow \neg D_{\forall R.D}(x) \vee C, \quad D_{\forall}(x) \vee (\forall R.D)(x)$$

Note that we do not have to apply this transformation on ABox clauses, as the role inversion rule of this calculus only applies to TBox clauses. In the next stage, all rules except the transitivity rule are applied. Observe that the transformation may introduce clauses that are not normal, as it is already the case for $Res_{\mathcal{ALCHI}}$ (see Section 5.3).

Example 7.1.4. We continue on the clause set $\mathcal{N}_9$ from Example 7.1.3 and compute inferences on B.

1. $\neg A(x) \vee (\forall r.D_1)(x)$
2. $\neg D_1(x) \vee B(x)$
3. $\neg D_1(x) \vee C(x)$
4. $r(a, b)$
5. $s(a, b)$
6. $\neg A(b) \vee \neg B(b)$
7. $\neg A(a) \vee D_1(b)$ *(Role Instantiation 1, 4, $\sigma = [x/a]$)*
8. $\neg A(a) \vee B(b)$ *(Resolution 2, 7, $\sigma = [x/b]$)*
9. $\neg A(a) \vee A(b)$ *(Resolution 6, 8, $\sigma = [b/b]$)*

7.2 Refutational Completeness

In order to show refutational completeness of $Int_{\mathcal{SHI}_\vee}$, we extend the model construction used for $Res^s_{\mathcal{ALCHI}}$ to incorporate derived ABox clauses. Let $\mathcal{N}$ be any set of $\mathcal{SHI}_\vee$ clauses such that $\bot \notin Res_{\mathcal{SHI}_\vee}(\mathcal{N})$, and denote the result of saturating $Res_{\mathcal{SHI}_\vee}(\mathcal{N})$ using the $\exists$-monotonicity and role assertion monotonicity rules by $\mathcal{N}^*$.

Denote by $\mathcal{N}_T^*$ the set of TBox clauses in $\mathcal{N}^*$, and by $\mathcal{N}_A^*$ the set of ABox clauses in $\mathcal{N}^*$. Using the model construction described in Section 4.2.3 and its modifications for inverse roles presented in Section 5.3, we can build a model for $\mathcal{N}_T^*$, which we denote by $\mathcal{I}_T = \langle \Delta^{\mathcal{I}_T}, \cdot^{\mathcal{I}_T} \rangle$.

We extend $\mathcal{I}_T$ to a full model for $\mathcal{N}$ using a construction solely based on ABox clauses. To ensure that TBox clauses are satisfied as well, we extend $\mathcal{N}_A^*$ by instantiating all TBox clauses.

$$\mathcal{N}_{A2}^* = \mathcal{N}_A^* \cup \{C[x/a] \mid a \in N_i, C \in \mathcal{N}_T^*\}$$

Lemma 7.2.1. *$\mathcal{N}_{A2}^*$ is saturated with respect to $Res_{\mathcal{SHI}_\vee}$.*

Proof. Note that we only unify two individual names if they are the same. The same is true for variables, since there is only one variable in our normal form. Accordingly, for every new pair of clauses $C_1[x/a], C_2[x/a] \in (\mathcal{N}_{A2}^* \setminus \mathcal{N}_A^*)$ on which a rule is applicable with the unifier $\sigma = [a/a]$, the same rule is applicable to C_1 and C_2 with $\sigma = [x/x]$. Accordingly, $\mathcal{N}_T^*$ contains the conclusion C_3. $C_3[x/a]$ is the corresponding conclusion on $C_1[x/a]$ and $C_2[x/a]$, and this clause is included in $\mathcal{N}_{A2}^*$ by construction. In the same way it can be argued for conclusions of inferences where one premise is a TBox clause and the other is an ABox clause. □

Let $\prec_l$ be a total ordering on concept literals satisfying the same constraints as the ordering used in the model construction for $Res_{\mathcal{ALC}}$ (see Section 4.2.3, page 65). Let $\prec_i$ be any total ordering on N_i. $\prec_l$ is extended to a total ordering on ABox literals by setting $L_1(a) \prec_l L_2(b)$, if $L_1 \prec_l L_2$ for all $a, b \in N_i$, and $L_1(a) \prec_l L_1(b)$, if $a \prec_i b$. An ABox literal L is *maximal* in an ABox clause C iff $L' \prec_l L$ for all ABox literals $L' \in C \setminus \{L\}$. Because $\prec_l$ is a total ordering, every clause has a unique maximal literal.

The ordering $\prec_c$ on ABox clauses is the multiset extension $(\prec_l)_{mul}$ of $\prec_l$, that is, $C_1 \prec_c C_2$ iff $C_1 \neq C_2$ and for every ABox literal $L_1 \in C_1$, $L_1 \prec_l L_2$ for some ABox literal $L_2 \in C_2$. The ordering makes it possible to enumerate the ABox clauses in $\mathcal{N}_{A2}^*$. In the following, C_i denotes the ith ABox clause in $\mathcal{N}_{A2}^*$ following this ordering. This means, if there are two clauses C_i and C_k with $i < k$, then also $C_i \prec_c C_k$.

We now complete the model $\mathcal{I}_T = \langle \Delta^{\mathcal{I}_T}, \cdot^{\mathcal{I}_T} \rangle$ of the TBox clauses in $\mathcal{N}_T^*$ to a model $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ of all clauses in $\mathcal{N}_T^* \cup \mathcal{N}_{A2}^*$. The domain $\Delta^{\mathcal{I}_T}$ is extended by one element x_a for every individual: $\Delta^{\mathcal{I}} = \Delta^{\mathcal{I}_T} \cup \{x_a \mid a \in N_i\}$. The interpretation function $\cdot^{\mathcal{I}}$ is

defined incrementally as follows, where $\mathcal{I}_i = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}_i} \rangle$. Note that x_D refers to the domain element assigned to the definer D, as introduced in the model construction in Section 4.2.3, and that the index i in C_i here only follows the enumeration on the ABox clauses in $\mathcal{N}^*_{A2}$.

1. $\cdot^{\mathcal{I}_0}$ is equal to $\cdot^{\mathcal{I}_T}$, except that for every role $r \in N_r$ and every individual $a \in N_i$, we set:

$$r^{\mathcal{I}_0} = r^{\mathcal{I}^T} \cup \{(x_a, x_b) \mid r(a,b) \in \mathcal{N}^* \text{ or } r^-(b,a) \in \mathcal{N}^*\}$$
$$a^{\mathcal{I}_0} = x_a$$

2. If $\mathcal{I}_i \models C_i$, then $\cdot^{\mathcal{I}_{i+1}} = \cdot^{\mathcal{I}_i}$. Otherwise:

   (a) If the maximal literal in C_i is of the form $A(a)$, $\cdot^{\mathcal{I}_{i+1}}$ is equal to $\cdot^{\mathcal{I}_i}$, except that $A^{\mathcal{I}_{i+1}} = A^{\mathcal{I}_i} \cup \{x_a\}$.

   (b) If the maximal literal in C_i is of the form $(\exists r.D)(a)$, $\cdot^{\mathcal{I}_{i+1}}$ is equal to $\cdot^{\mathcal{I}_i}$, except that $r^{\mathcal{I}_{i+1}} = r^{\mathcal{I}_i} \cup \{(x_a, x_D)\}$.

   (c) If the maximal literal of C_i is of the form $(\exists r^-.D)(a)$, $\cdot^{\mathcal{I}_{i+1}}$ is equal to $\cdot^{\mathcal{I}_i}$, except that $r^{\mathcal{I}_{i+1}} = r^{\mathcal{I}_i} \cup \{(x_D, x_a)\}$.

3. $\cdot^{\mathcal{I}} = \cdot^{\mathcal{I}_n}$, where n is the number of ABox clauses in $\mathcal{N}^*_{A2}$.

For Steps 2b and 2c, observe that for each maximal $\exists R.D$ in a clause C_i with $\mathcal{I}_i \not\models C_i$, there is always a corresponding domain element x_D. A definer D only has a corresponding domain element in the candidate model if $\neg D(x) \notin \mathcal{N}^*$. If $\neg D(x) \in \mathcal{N}^*$, the $\exists$-elimination rule applies on all literals of the form $\exists R.D$, such that these literals are not taken into account by the model construction (see also Lemma 4.2.16).

Similarly to Lemmata 4.2.13 and 4.2.14, we prove that the model construction for $\mathcal{SHI}_\vee$ knowledge bases is monotone.

Lemma 7.2.2. *For every ABox clause C_i, $\mathcal{I}_{i+1} \models C_i$ implies $\mathcal{I} \models C_i$.*

Proof. If $\mathcal{I}_{i+1} \models C_i$, there is a literal $L \in C_i$ such that $\mathcal{I}_{i+1} \models L$. We distinguish the cases for L, and show that in each case $\mathcal{I} \models L$.

1. L is a positive literal of the form $A(a)$ or $(\exists R.D)(a)$. Since the model construction only adds values to the interpretation function, but does not remove any, $\mathcal{I} \models L$.

2. L is of the form $\neg A(a)$. $A(a)$ cannot be a positive and maximal literal in any clause C_j larger than C_i. Therefore, it is impossible that x_a is added to $A_j^{\mathcal{I}}$ for any index j larger than i, and $\mathcal{I} \models L$.

3. L is of the form $(\forall R.D)(a)$. Due to the constraints of the ordering, there can be no clause C_j with $C_i \prec C_j$ in which the maximal literal is of the form $(\exists R.D')(a)$. Therefore, $\mathcal{I} \models L$.

We obtain that in each case, $\mathcal{I}_{i+1} \models C_i$ implies $\mathcal{I} \models C_i$. □

Lemma 7.2.3. *For every ABox clause C_i, $\mathcal{I}_{i+1} \not\models C_i$ implies $\mathcal{I} \not\models C_i$.*

Proof. $\mathcal{I}_{i+1} \not\models C_i$ is only possible if the maximal literal L in C_i is of the form $\neg A(a)$ or $(\forall R.D)(a)$, since the other cases are taken care of by the model construction. We distinguish both cases.

1. L is of the form $\neg A(a)$. Then, $a^{\mathcal{I}_{i+1}} \in A^{\mathcal{I}_{i+1}}$. Since no step in the model construction removes elements from the interpretation, $a^{\mathcal{I}} \in A^{\mathcal{I}}$, and hence $\mathcal{I} \not\models C_i$.

2. L is of the form $(\forall R.D)(a)$. If $\mathcal{I}_{i+1} \not\models (\forall R.D)(a)$, there is a domain element $x \in \Delta^{\mathcal{I}}$ such that $(a, x) \in R^{\mathcal{I}_{i+1}}$ and $x \notin D^{\mathcal{I}_{i+1}}$. If in addition $\mathcal{I} \models (\forall R.D)(a)$, we must have $x \in D^{\mathcal{I}}$. This is only possible if there is a clause C_j with $C_i \prec C_j$ in which the maximal literal is of the form $D(b)$, which is impossible due to the constraints of the ordering. Therefore, $\mathcal{I} \not\models (\forall R.D)(a)$ implies $\mathcal{I} \not\models (\forall R.D)(a)$.

We obtain that in each case, $\mathcal{I}_{i+1} \not\models C_i$ implies $\mathcal{I} \not\models C_i$. □

We can now prove that $\mathcal{I}$ satisfies all ABox clauses C_i, which establishes that $\mathcal{I}$ is a model of $\mathcal{N}_{A2}^*$.

Lemma 7.2.4. *For every ABox clause C_i in $\mathcal{N}_{A2}^*$, $\mathcal{I} \models C_i$.*

Proof. The proof is by contradiction. Let C_i be the smallest ABox clause according to $\prec_c$ such that $\mathcal{I} \not\models C_i$. We distinguish the different cases for the maximal literal L in C_i, where $C_i = L \vee C_i'$. Since $\mathcal{I} \not\models C_i$, also $\mathcal{I} \not\models L$.

1. $L = A(a)$. Then $\mathcal{I}_{i+1} \models C_i$ due to Step 2a of the model construction, and $\mathcal{I} \models C_i$ due to Lemma 7.2.2, which contradicts our assumption.

2. $L = \neg A(a)$. Since $\mathcal{I} \not\models L$, $x_a \in A^{\mathcal{I}}$. If $x_a \in A^{\mathcal{I}}$, due to Step 2a of the model construction, there is a smaller clause $C_j = A(a) \vee C'_j$, such that $A(a)$ is maximal in C_j and $\mathcal{I}_j \not\models C'_j$. But then, due to the resolution rule, there is also the clause $C_k = C'_j \sqcup C'_i$. $k < j$, since $A(a)$ is maximal in C_j, $\neg A(a)$ is maximal in C_i, and $A(a) \prec_l \neg A(a)$. Since $\mathcal{I}_i \not\models C_i$ and $\mathcal{I}_j \not\models C_j$, $\mathcal{I}_{k+1} \not\models C_k$, and due Lemma 7.2.3, also $\mathcal{I} \not\models C_k$. But this contradicts the initial assumption that C_i is the smallest clause with $\mathcal{I} \not\models C_i$.

3. $L = (\exists R.D)(a)$. Then $a^{\mathcal{I}} \in (\exists R.D)^{\mathcal{I}}$ due to Lemma 7.2.2 and Steps 2b and 2c of the model construction.

4. $L = (\forall R.D)(a)$. This means, there is an edge $(x_a, x') \in R^{\mathcal{I}}$ such that $x' \notin D^{\mathcal{I}}$. There are two possibilities for why $x' \notin D^{\mathcal{I}}$.

   (a) There is a role assertion $R(a, b) \in \mathcal{N}^*$ and $x' = x_b$. But then, the role assertion instantiation rule applies between $R(a, b)$ and C_i, and the clause $C_j = C'_i \vee D(b)$ is inferred. Since $\mathcal{I} \not\models C_i$, also $\mathcal{I} \not\models C'_i$, and since $x_b \notin D^{\mathcal{I}}$, also $\mathcal{I} \not\models C_j$. But in the ordering, $D(b)$ is smaller than $(\forall R.D)(a)$, and therefore $C_j \prec_c C_i$. This contradicts the initial assumption that C_i is the smallest clause that is not satisfied by the model.

   (b) There is a clause $C_j = (\exists R.D_1)(a) \vee C'_j$, in which $(\exists R.D_1)(a)$ is maximal, $D_1 \neq D$, $\neg D_1(x) \vee D(x) \notin \mathcal{N}^*$ and $\mathcal{I}_j \not\models C_j$. Due to the role propagation rule, we also have the clause $C_k = C'_i \vee C'_j \vee (\exists R.D_2)(a)$, together with the two clauses $\neg D_2(x) \vee D(x)$ and $\neg D_2(x) \vee D_1(x)$. Since $(\exists R.D_2)(a) \prec_l (\forall R.D)(a)$ and $C'_j \prec C_i$, C_k is smaller than C_i. We also have $\mathcal{I} \not\models C'_i$ and $\mathcal{I} \not\models C'_j$. There are two possibilities.

      i. If $k < j$, since both $\mathcal{I}_i \not\models C'_i$ and $\mathcal{I}_j \not\models (\exists R.D_1)(a)$, we must have $\mathcal{I}_j \not\models (\exists R.D_2)(a)$. But then, $\mathcal{I}_j \not\models C_k$ and $\mathcal{I} \not\models C_k$, which contradicts the assumption that C_i is the smallest clause in $\mathcal{N}^*_{A2}$ that is not satisfied by the model.

      ii. $j < k$. This means there are literals in C_k that are larger than all literals in C_j. This implies that $(\exists R.D_2)(a)$ is not maximal in C_k. If $(\exists R.D_2)(a)$ is not maximal in C_k, Step 2b of the model construction

does not result in $\mathcal{I}_{k+1} \models (\exists R.D_2)(a)$, and since also $\mathcal{I} \not\models C'_j$ and $\mathcal{I} \not\models C'_i$, $\mathcal{I}_{k+1} \not\models (C'_j \cup C'_i)$. But then, due to Lemma 7.2.3, no literal in C_k is satisfied in $\mathcal{I}$, which contradicts our initial assumption that C_i is the smallest clause that is not satisfied by $\mathcal{I}$.

Every case leads to a contradiction. Hence, there can be no smallest ABox clause $C \in \mathcal{N}^*_{A2}$ such that $\mathcal{I} \not\models C$. Therefore, $\mathcal{I} \models C$ for all ABox clauses in $C \in \mathcal{N}^*_{A2}$. □

Lemma 7.2.5. *For every TBox clause in $\mathcal{N}^*_T$, $\mathcal{I} \models C$.*

Proof. The proof is by contradiction. Assume C is the smallest TBox clause in $\mathcal{N}^*_T$ such that $\mathcal{I} \not\models C$. Then, there must be a domain element $x \in \Delta^{\mathcal{I}}$ such that $x \notin C^{\mathcal{I}}$. x is either a domain element from $\Delta^{\mathcal{I}_T}$, or it is a domain element that represents an individual. We distinguish these two cases.

1. $x \notin \Delta^{\mathcal{I}_T}$. Then there is an individual $a \in N_i$ such that $x = a^{\mathcal{I}}$. $\mathcal{N}^*_{A2}$ contains an ABox clause C_A with $C_A = C[x/a]$ due to construction. By Lemma 7.2.4, $\mathcal{I} \models C_A$. Since C_A contains a as only individual, this implies $a^{\mathcal{I}} \in C^{\mathcal{I}}$. But this contradicts the initial assumption that $x \notin C^{\mathcal{I}}$.

2. $x \in \Delta^{\mathcal{I}_T}$. Then $x \in C^{\mathcal{I}_T}$ by Lemma 5.6.1. The only reason $x \notin C^{\mathcal{I}}$ can be true is due to an additional edge to x that has been added by the model construction. Assume this edge is $(x, a^{\mathcal{I}}) \in R^{\mathcal{I}}$. This edge is the reason for $x \notin C^{\mathcal{I}}$, and hence C must contain a literal of the form $(\forall R.D_1)(x)$. Since $(x, a^{\mathcal{I}}) \in R^{\mathcal{I}}$, there is an ABox clause C_i of which the maximal literal is of the form $(\exists S.D_2)(a)$ and $\mathcal{I}_{i-1} \not\models C_i$, where $\mathsf{Inv}(S) \sqsubseteq_{\mathcal{N}} R$ and D_2 is the corresponding definer for x. Suppose $C = C' \vee (\forall R.D_1)(x)$ and $C_i = C'_i \vee (\exists S.D_2)(a)$.

   Note that we do the same additional normal form transformation step for clauses with universal restrictions for $Res_{\mathcal{ALCHI}}$ as we do for $Res_{\mathcal{SHI}_\vee}$. For this reason, Lemma 5.3.2, which has been formulated for $Res_{\mathcal{ALCHI}}$, also applies to $Res_{\mathcal{SHI}_\vee}$. Lemma 5.3.2 states that if $C \vee (\forall R.D_1)(x) \in \mathcal{N}^*$, either $C = D_{\forall R.D_1}(x)$, or there are the clauses $D_{\forall R.D_1}(x) \vee (\forall R.D_1)(x)$, $\neg D_{\forall R.D_1}(x) \vee C' \in \mathcal{N}^*$. Since $x \notin (C')^{\mathcal{I}}$, $x \notin (D_{\forall R.D_1})^{\mathcal{I}}$.

   The role inversion rule infers from $D_{\forall R.D_1}(x) \vee (\forall R.D_1)(x)$ the clause $D_1(x) \vee (\forall \mathsf{Inv}(R).D_{\forall R.D_1})(x)$. Since $S \sqsubseteq_{\mathcal{N}} \mathsf{Inv}(R)$, we can apply the $\forall\exists$-role propagation

rule on this clause and C_i, resulting in the clause $C_j = C_i' \vee D_1(a) \vee (\exists S.D_3)(a)$, where D_3 represents $D_2 \sqcap D_{\forall R.D_1}$. Because $x \notin (D_{\forall R.D_1})^{\mathcal{I}}$, also $x \notin D_3^{\mathcal{I}}$. Observe that $C_j \prec_c C_i$. Since $\mathcal{I}^{i-1} \not\models C_i$, we cannot have $\mathcal{I}^{i-1} \models (\exists S.D_3)(a)$, since this would imply $\mathcal{I}^{i-1} \models (\exists S.D_2)(a)$. We also have $\mathcal{I} \not\models C_i'$, as observed earlier. Because C_j is an ABox clause, $\mathcal{I} \models C_j$ by Lemma 7.2.4. The only literal left in C_j that can be satisfied by $\mathcal{I}$ is therefore $D_1(a)$. We obtain that $a^{\mathcal{I}} \in D_1^{\mathcal{I}}$, which contradicts the assumption that $(x, a^{\mathcal{I}}) \in R^{\mathcal{I}}$ serves as a counter example for $x \in (\forall R.D_1)^{\mathcal{I}}$. Therefore, $x \in (\forall R.D_1)^{\mathcal{I}}$, which contradicts the original assumption that C is not satisfied by the model.

In every case, the assumption that there is a domain element $x \in \Delta^{\mathcal{I}}$ with $x \notin C^{\mathcal{I}}$ leads to a contradiction. Therefore, $\mathcal{I} \models C$ for all TBox clauses. □

We have established that $\mathcal{I}$ is a model for $\mathcal{N}_T^* \cup \mathcal{N}_{A2}^*$. The only clauses in $\mathcal{N}^*$ that are left are the role axioms of the form $R \sqsubseteq S$ and $\mathsf{trans}(R)$. Observe that $\mathcal{N}^*$ is saturated using the $\exists$-role propagation rule and the role assertion monotonicity rule. This ensures that for every tuple $(x_1, x_2) \in R^{\mathcal{I}}$ and every role S with $R \sqsubseteq_{\mathcal{N}} S$, $(x_1, x_2) \in S^{\mathcal{I}}$ (see also Section 5.1.2). In order to construct a model for the transitivity axioms, we extend $\mathcal{I}$ to the interpretation $\mathcal{I}^{\mathsf{trans}}$, using the transformation described in Section 5.2, by adding the transitive closure to all roles that are transitive or subroles of a transitive role. The following lemma can be established in the same way as for $Res_{\mathcal{ALCH}}$ and $Res_{\mathcal{SH}}$.

Lemma 7.2.6. *Let $C \in \mathcal{N}^*$ be a role axiom of the form $\mathsf{trans}(R)$ or $R \sqsubseteq S$. Then, $\mathcal{I}^{\mathsf{trans}} \models C$.*

Note that it can be shown in the same way as for Lemma 5.2.4 that the additional edges in $\mathcal{I}^{\mathsf{trans}}$ have no effect on the satisfaction of universal role restrictions in $\mathcal{I}$. Together with the Lemmata 7.2.4, 7.2.5 and 7.2.6, we establish that all TBox clauses, ABox clauses and RBox axioms in $\mathcal{N}^*$ are satisfied by $\mathcal{I}^{\mathsf{trans}}$.

Lemma 7.2.7. *Let $\mathcal{N}$ be any set of $\mathcal{SHI}_\vee$ clauses. If $Res_{\mathcal{SHI}_\vee}(\mathcal{N})$ does not contain the empty clause, we can build a model for it.*

Theorem 7.2.8. *$Res_{\mathcal{SHI}_\vee}$ is sound, refutationally complete and terminating, and provides a decision procedure for satisfiability of $\mathcal{SHI}\nu_\vee$ knowledge bases.*

Proof. Since only finitely many definer symbols are introduced and clauses are represented as sets, the number of clauses that can be derived is bounded. This establishes termination of the calculus. The soundness of the rules can be argued in the same way as in for $\mathcal{SHI}$ (see Lemma 4.2.2, Lemma 4.2.3 and Theorem 5.3.4). Hence, if $\bot \in \mathcal{N}^*$, $\mathcal{N}$ is unsatisfiable. On the other hand, if $\bot \notin \mathcal{N}^*$, we can build a model for $\mathcal{N}$ (Lemma 7.2.7). Using our normal form transformation, we can transform any $\mathcal{SHI}\nu_\vee$ knowledge base into an equi-satisfiable set of clauses. Therefore, the calculus provides a sound and refutationally complete decision procedure for satisfiability of $\mathcal{SHI}\nu_\vee$ ontologies. □

Since the model construction only depends on the maximal literal in each clause, we can refine $Res_{\mathcal{SHI}_\vee}$ to a calculus that only performs inferences on maximal literals, using an arbitrary ordering on concept and role symbols. Moreover, all role combination rules are adaptions of corresponding rules in $Int_{\mathcal{SHI}}$. Therefore, we can argue in the same way as for $Int_{\mathcal{SH}}$ and $Int_{\mathcal{ALCHI}}$ that $Int_{\mathcal{SHI}_\vee}$ is interpolation complete for $\mathcal{SHI}\nu_\vee$.

Theorem 7.2.9. *$Int_{\mathcal{SHI}_\vee}$ is interpolation complete for $\mathcal{SHI}\nu$ ontologies with disjunctive ABoxes.*

7.3 Representing the Result in $\mathcal{SHOI}$

$Int_{\mathcal{SHI}_\vee}$ can be used for computing a finite uniform interpolant of any $\mathcal{SHI}$ knowledge bases, but in an extended language, because these uniform interpolants may use greatest fixpoint operators and disjunctive concept assertions. Since some applications require the uniform interpolant to be compatible with standard description logic reasoners or the web ontology standard language OWL, it is of interest to represent the uniform interpolant in a more common description logic. A representation without fixpoints can be obtained by approximation or the usage of helper concepts using the technique described in Section 4.1.3.

Similarly, using a technique presented in Areces et al. (2003), we can represent arbitrary disjunctive ABoxes by classical ABoxes using additional roles. Since this might not always be desirable, we introduce a technique that can be used to approximate disjunctive ABoxes in such a way that the uniform interpolant can be fully represented

in the signature using only classical description logic constructs. This transformation may result in the use of nominals in the ontology.

The following definition describes the condition under which we can directly transform a disjunctive concept assertion into a classical concept assertion.

Definition 7.3.1. Given two individuals a_1 and a_2 and a disjunctive ABox $\mathcal{A}$, a_1 and a_2 are *connected in* $\mathcal{A}$ if a_1 and a_2 are connected in the graph corresponding to the role assertions in $\mathcal{A}$. A disjunctive concept assertion α is *connected in* $\mathcal{A}$ if every pair of individuals in α is connected in $\mathcal{A}$.

Note that concept assertions with one individual are always connected. Let $\mathcal{A}$ be an ABox and $\alpha = C_1(a) \vee C_2(b) \vee \alpha'$ be a disjunctive concept assertion that is connected in $\mathcal{A}$, where α may be empty. Then, there is a sequence of role assertions $R_1(a_1, a_2)$, $R_2(a_2, a_3)$, ..., $R_n(a_n, a_{n+1})$ such that $a = a_1$, $b = a_{n+1}$ and $\mathcal{A} \models R_i(a_i, a_{i+1})$ for each i with $1 \leq i \leq n$. Denote by α_2 the following disjunctive concept assertion.

$$(C_1 \sqcup \exists R_1.\exists R_2. \ldots . \exists R_n(C_2 \sqcap \{b\}))(a) \vee \alpha'$$

It is easy to verify that $\mathcal{A}_2 = (\mathcal{A} \setminus \alpha) \cup \{\alpha_2\}$ is logically equivalent to $\mathcal{A}$.

Using this technique, we can replace all connected disjunctive concept assertions in $\mathcal{A}$ by classical concept assertions. If a disjunctive concept assertion α is not connected in $\mathcal{A}$, we cannot express α as a classical concept assertion, but α might still contribute to the entailment of other classical concept assertions. To compute a set of clauses that can be fully translated into $\mathcal{SHOI}\nu$, and that preserves all entailments that are classical $\mathcal{SHOI}$ axioms, we use our calculus in a similar way as for computing uniform interpolants. Let $\mathcal{N}$ be any set of clauses and $\mathcal{N}^*$ the saturation of $\mathcal{N}$. The set $\mathcal{N}^{conv}$ is the smallest set of clauses $C \in \mathcal{N}^*$ such that every ABox clause $C \in \mathcal{N}^*$ is connected and that includes all clauses $C \in \mathcal{N}^*$ that additionally satisfy at least one of the following conditions:

1. $C \in \mathcal{N}$.
2. C is the conclusion of any rule application on a clause that is not connected in $\mathcal{N}^*$.

Note the similarity of this definition and Definition 4.4.12 defining minimal clausal representations of uniform interpolants. $\mathcal{N}^{conv}$ preserves all entailments of $\mathcal{N}$ that are

representable as classical $\mathcal{SHOI}$ axioms. $\mathcal{N}^{conv}$ can be transformed into a $\mathcal{SHOI}\nu$ knowledge base without definers, in which all remaining disjunctive concept assertions are represented as classical concept assertions using nominals.

Theorem 7.3.2. *Let $\mathcal{N}$ be any knowledge base in normal form. Then, the described method for approximating disjunctive concept assertions computes a $\mathcal{SHOI}\nu$ knowledge base $\mathcal{K}_{\mathcal{SHOI}\nu}$, and, for any classical $\mathcal{SHI}$ axiom α without definers, that $\mathcal{N} \models \alpha$ iff $\mathcal{K}_{\mathcal{SHOI}\nu} \models \alpha$.*

Proof. Since our normal form transformation preserves all entailments modulo definers, we can do the proof on the normal form representation of each involved knowledge base. Denote by $\mathcal{N}$ the input clause set, and by $\mathcal{N}^{conv}$ the approximated clause set that only contains connected clauses. Let α be any $\mathcal{SHI}$ axiom. We show that $\mathcal{N} \models \alpha$ iff $\mathcal{N}^{conv} \models \alpha$.

If $\mathcal{N}$ is unsatisfiable, this is trivially true. Assume $\mathcal{N}$ is satisfiable. Since $\mathcal{N}^{conv}$ consists only of clauses inferred from $\mathcal{N}$, $\mathcal{N}^{conv} \models \alpha$ implies $\mathcal{N} \models \alpha$. We therefore only have to show that $\mathcal{N} \models \alpha$ implies $\mathcal{N}^{conv} \models \alpha$. Assume $\mathcal{N} \models \alpha$. If α is a TBox axiom, $\mathcal{N}^{conv} \models \alpha$, since the approximation only affects ABox axioms. The same is the case if α is a role assertion. Assume therefore that α is a concept assertion of the form $C(a)$.

$\mathcal{N} \models C(a)$ iff $\mathcal{N} \cup \{\neg C(a)\} \models \bot$. Let $\mathcal{M}$ be the normal form representation of $C(a)$. We show that whenever $\mathcal{N} \cup \mathcal{M} \models \bot$, also $\mathcal{N}^{conv} \cup \mathcal{M} \models \bot$. $\mathcal{M}$ has the following properties:

1. Every TBox clause in $\mathcal{M}$ is of the form $\neg D(x) \vee C$, where D does not occur in $\mathcal{N}$ or $\mathcal{N}^{conv}$.

2. Every ABox clause only contains the individual a occurring in $C(a)$.

Despite the second property, it is possible that clauses in $\mathcal{M}$ contribute to the derivation of ABox clauses containing more than one individual. This is possible either due to resolution on clauses from $\mathcal{N}$ that have more than one individual, or due to the role instantiation rule. More precisely, if $\mathcal{M}$ contains a clause of the form $(\forall r.D)(a) \vee C$, and $\mathcal{N}$ contains a role assertion $r(a, b)$, $D(b) \vee C$ is derived. Inferences of this sort are however only possible through role assertions containing a. Let N_i^a denote all individuals in $\mathcal{N}$ that are connected to a, where a is the sole individual occurring

in $\mathcal{M}$. We define an ordering $\prec_i$ on individuals such that for each pair of individuals a_1, a_2 with $a_1 \in N_i^a$ and $a_2 \notin N_i^a$, we have $a_1 \prec_i a_2$. Let $\prec_l$ be an ordering on ABox literals that fulfils all constraints given in Section 7.1 such that $C(a_1) \prec_l C(a_2)$ iff $a_1 \prec_i a_2$. $Res^{norm\prec_l}_{\mathcal{SHI}_\vee}$, the refinement of $Res_{\mathcal{SHI}_\vee}$, is sound and refutationally complete. $Res^{norm\prec_l}_{\mathcal{SHI}_\vee}$ gives precedence to ABox literals with an individual that is not connected to a, and only infers clauses that are normal.

In connected clauses, all individuals are connected to each other. Every clause C that is not connected must contain at least one literal $L(a')$ with $a' \notin N_i^a$, which is maximal in C. Therefore, the only ABox clauses on which an inference with a non-connected clause can be performed with $Res^{\prec_l}_{\mathcal{SHI}_\vee}$ are clauses that are either in $\mathcal{N}$ or are inferred using clauses in $\mathcal{N}$. Moreover, every TBox clause in $\mathcal{M}$ contains at least one negative definer literal. Due to the side conditions of the rules, no inference between such a clause and an ABox clause is possible, unless the ABox clause contains a positive definer literal of the form $D(a)$, since the conclusion would otherwise contain an ABox literal of the form $\neg D(a)$. Since $\mathcal{M}$ does not share any definer symbols with $\mathcal{N}$, this means that no rule application is possible on a clause from $\mathcal{M}$ and a clause that is not connected.

Therefore, if the empty clause can be derived from $\mathcal{N} \cup \mathcal{M}$, and if the derivation involves a clause C that is not connected, then this derivation also involves a clause C' that can be derived from C using only clauses in $\mathcal{N}$. But these clauses are all included in $\mathcal{N}^{conv}$. Hence, if we can derive the empty clause from $\mathcal{N} \cup \mathcal{M}$, then we can also derive it from $\mathcal{N}^{conv} \cup \mathcal{M}$.

We establish that for any classical $\mathcal{SHI}$ axiom α that does not use definer symbols, $\mathcal{N} \models \alpha$ iff $\mathcal{N}^{conv} \models \alpha$, and $\mathcal{N} \models \alpha$ iff $\mathcal{K}_{\mathcal{SHOI}\nu} \models \alpha$. □

This result enables us to compute representations of uniform interpolants of $\mathcal{SHI}$ knowledge bases in the classical description logic $\mathcal{SHOI}$. Using the method described in the last section, we can always compute a uniform interpolant in $\mathcal{SHI}\nu_\vee$. Using the method described in this section, this uniform interpolant can be approximated in $\mathcal{SHOI}\nu$, and the approximation is actually a uniform interpolant of the input. In order to obtain a representation in $\mathcal{SHOI}$, we use one of the techniques described in Section 4.1.3.

Chapter 8

Implementation and Evaluation

The main topic of the thesis is the development of practical methods for uniform interpolation in expressive description logics. In the previous chapters, we presented uniform interpolation methods for expressive description logics that are based on resolution and saturation. The motivation for using these techniques is to obtain practical methods for the computation of uniform interpolants. In this chapter, we show that these approaches indeed allow for practical implementations that are able to compute uniform interpolants of realistic ontologies for different types of signatures.

We implemented three prototypes of different uniform interpolation methods presented in the thesis: (1) uniform interpolation of $\mathcal{ALCH}$ ontologies, using the calculus presented in Section 5.1, (2) forgetting concept symbols from $\mathcal{SHQ}$ ontologies, using the calculus presented in Chapter 6, and (3) uniform interpolation of $\mathcal{ALC}$ knowledge bases with ABoxes, which uses a restricted version of the method presented in Section 7. The main ideas used in the implementations are discussed in Section 8.1.

We evaluated these prototypes on a corpus extracted from the NCBO BioPortal repository, which contains real-life ontologies and knowledge bases from bio-medical applications. This corpus is described Section 8.2. For the evaluation itself, we undertook three experimental studies: (1) forgetting small sets of symbols from an ontology or knowledge base, (discussed in Section 8.3), (2) computing uniform interpolants for small signatures (discussed in Section 8.4), and (3) computing uniform interpolants for central symbols of the ontology (discussed in Section 8.5).

The experimental studies are motivated by the following considerations.

(1) *They allow for a practical evaluation.* In order to obtain statistically significant

results, it is important to choose sample sizes that are not too small. Our corpus contains 306 ontologies, and we chose a sample size of 360 for each signature size. For applications in information hiding and ontology reuse, where the uniform interpolant is not computed frequently, computation times of several hours are acceptable. However, in order to be able to make a complete evaluation on all samples and ontologies in reasonable time, it is necessary to select a smaller time frame for each computation. This is only possible if we restrict ourselves to easier signatures.

(2) *They directly correspond to applications mentioned in the introduction.* For instance, in order to compute the logical difference between subsequent versions of an ontology, it is to be expected that only few symbols have to be eliminated. For ontology analysis on the other hand, it makes most sense to compute uniform interpolants for small signatures, in order to get a refined view of the relations between the symbols in these signatures. Finally, computing a uniform interpolant for a signature of central symbols corresponds to the application of ontology summary.

(3) Finally, *they give an idea of the practicality of uniform interpolation for other signatures.* Signatures that are small, respectively large, with respect to the signature of the input ontology are two extremes in the range of possible signatures. In each case, we chose signatures of different size, to get an idea of how the performance changes with changing signatures. As it turns out, the task becomes more difficult towards both ends: on average, forgetting 50 symbols is easier than forgetting 100 symbols, and forgetting all but 50 symbols is easier than forgetting all but 100 symbols. But our results indicate that the difficulty not only depends on the size of the signature, but also on to the symbols that occur in the signature. Most ontologies contain a small set of symbols that are central to that ontology and harder to eliminate. When forgetting larger sets of symbols, the probability of eliminating one of these central symbols increases. On the other hand, when computing uniform interpolants for very small signatures, the probability that the central symbols are connected to the selected signature is smaller than if the signature is larger. To support this hypothesis, in Section 8.5 we computed uniform interpolants for signatures that could be considered central to the ontology.

Our results indicate that, while it might not be possible to compute uniform interpolants for any given signature for all ontologies, by choosing the signature wisely and

integrating what is central to the ontology, it might also be possible to compute uniform interpolants of for larger and smaller signature sizes than used in the evaluation. We believe that this is not a major restriction for most applications. For example, a uniform interpolant generated for ontology reuse would typically use terms that are central to the ontology. In information hiding, it should generally not be the case that the confidential information in an ontology is also the most central information in the ontology. For ontology obfuscation, it might be necessary to eliminate symbols that are central to the ontology, but a small number of these symbols might be sufficient for the task, as was illustrated in Ludwig and Konev (2013, 2014).

8.1 Implementation

8.1.1 Overview of the Algorithm

We implemented three different prototypes for the calculi for $\mathcal{ALCH}$ ontologies, $\mathcal{SHQ}$ ontologies and $\mathcal{ALC}$ knowledge bases. The prototype for $\mathcal{SHQ}$ can only be used for forgetting concept symbols, as our method for $\mathcal{SHQ}$ is not able to eliminate role symbols (see Chapter 6). The prototypes were implemented in the multi-paradigm programming language Scala (Odersky et al., 2004). We further used the OWL-API (Horridge and Bechhofer, 2011) for various tasks such as parsing, reasoning and module extraction (see below), even though in the main algorithms we used own data structures. The prototype also makes use of reasoning services provided by the description logic reasoner HermiT (Glimm et al., 2014). Scala is fully compatible with Java, which means the implemented prototypes are not only useful for the evaluation presented in this chapter, but can also directly be used as libraries for platform independent Java applications. The prototypes are made available as the tool and library Lethe (Koopmann and Schmidt, 2015c), which also provides functionality for abductive reasoning based on the uniform interpolation calculi, which are not discussed in the thesis.

We reduce the problem of uniform interpolation for arbitrary signatures to the problem of forgetting singular symbols, which we process one by one. The top level algorithm in each prototype is as follows.

1. Preprocess the input ontology.

2. For each symbol to be forgotten:
   - Apply the corresponding forgetting algorithm.
3. Postprocess the resulting ontology.

Preprocessing does not involve the clausification, but various optimisations described in Section 8.1.2. Further pre- and postprocessing is applied when a particular symbol is eliminated. As a result of the preprocessing, some symbols might already be removed from the ontology. After the preprocessing step, we determine an order in which symbols are eliminated from the ontology. We found that a simple, but efficient heuristic for eliminating symbols is to process them ordered by frequency, starting from the symbols that occur less often in the ontology. These symbols can usually be eliminated very quickly, and often reduce the set of clauses to be processed for subsequent symbols. For very frequently occurring symbols the effect was found to be the opposite. For each symbol s, we apply the corresponding forgetting algorithm only on the subset of the ontology that contains s, which restricts the scope of clausification, redundancy elimination and optimisation techniques.

Depending on whether a concept or a role symbol is eliminated, we use a different algorithm. The general structure of these algorithms is the same, but the reasoning is implemented differently. The forgetting algorithm is described in Sections 8.1.3–8.1.5, where we describe a specific approach for forgetting roles, as well as the search strategies used by the prototypes.

8.1.2 Pre- and Postprocessing

As first step in the preprocessing, we use syntactical module extraction to reduce the set of axioms in the ontology. A module is a subset of an ontology that, together with other properties, preserves all entailments over a specified signature. $\top\bot*$-modules are an approximation of minimal modules that can be computed cheaply on a syntactic level (Sattler et al., 2009, see also Section 2.6). The $\top\bot*$-module preserves all entailments in the desired signature, but usually contains additional symbols. In some instances, the modules still cover a substantial part of the input ontology. In the majority of cases however, module extraction leads to a significant reduction of the number of axioms to be processed by our prototypes.

After a module is extracted, we apply purification on the declausified ontology to eliminate initial symbols from the ontology. For this, we determine which concept symbols to be eliminated occur only positively or only negatively in the ontology. If a concept symbol occurs only positively, we replace it by $\top$. If a concept symbol occurs only negatively, we replace it by $\bot$.

If a concept symbol occurs only positively or only negatively in an ontology, no resolution steps are possible on the clausal representation, and the forgetting method would simply delete all clauses that contain this symbol. This has the logically equivalent effect of replacing positively occurring symbols by $\top$ and negatively occurring symbols by $\bot$. Purification does not require any preceding normalisation of the input. Moreover, it can be performed in linear time, since the ontology has to be processed only twice: first to determine the polarities of symbols, and a second time for rewriting the ontology.

Another optimisation, inspired by Ludwig and Konev (2013, 2014), is to use further structural transformation on the ontology. More precisely, for every sub-expression C_s in concepts of the form $C_s \sqcup C$, where C_s does not contain the symbol we want to eliminate, we replace C_s by a new concept symbol X_s. After the symbol in question is eliminated, we replace each introduced symbol X_s by its corresponding concept C_s. This step significantly reduces the literals and clauses the algorithm has to process, especially if the axioms are further optimised beforehand. For instance, a sequence of clauses of the form $C_1 \sqcup A$, ..., $C_n \sqcup A$ can be represented by just a single clause $X \sqcup A$, given that the clauses all contain the same or all contain no negative definer literal. Furthermore, the number of role restrictions in the input ontology is reduced to the cases where the symbol to be forgotten actually appears under the role restriction, which helps in avoiding unnecessary role propagation steps, a problem that is discussed in detail in Sections 8.1.4 and 8.1.5.

A drawback of this additional structural transformation step is that it can hide redundancies, which would otherwise be eliminated by the subsumption deletion rule. A typical example are clauses representing disjointness axioms. These are usually binary clauses of the form $\neg A_1 \sqcup \neg A_2$, and can subsume a large number of inferred clauses. If we replace all disjointness axioms in the signature by a simple axiom of the form $\top \sqsubseteq X$, no subsumption deletion with these clauses is possible, and we may infer

a lot of redundant clauses. It is worth noting that this problem already occurs due to our preselection of axioms on which the calculus is applied, since we only select axioms that contain the symbol to be eliminated. For most ontologies however, we cannot avoid these redundancies, since without these optimisations the clause sets become too large to be processed in reasonable time.

A second optimisation influenced by Ludwig and Konev (2013, 2014) aims at reducing the number of role restrictions, which are the main source of the theoretical worst-case complexity of our algorithm (see Lemma 4.2.4). In particular, concepts of the form $(\forall r.C_1 \sqcap \forall r.C_2)$ are simplified to concepts of the form $\forall r.(C_1 \sqcap C_2)$ (this step basically applies the $\forall\forall$-role propagation rule on declausified axioms), and concepts of the form $\exists r.C_1 \sqcup \exists r.C_2$ are simplified to concepts of the form $\exists r.(C_1 \sqcup C_2)$. These simplifications are implemented using simple rewrite-rules.

Moreover, we perform some simple transformations based on unsatisfiable and tautological concepts, which are applied after a symbol has been eliminated. Concepts of the form $\forall r.\top$ and $\top \sqcup C$ are replaced by $\top$, and concepts of the form $\exists r.\bot$ and $\bot \sqcap C$ are replaced by $\bot$. When eliminating definer symbols, we use these syntactic transformations to determine whether the corresponding fixpoint expressions are tautological. If D is defined by the axiom $D \sqsubseteq C$, and $C[D/\top] \equiv \top$, we have that $\nu X.C[X] \equiv \top$ ($\top$ is the greatest fixpoint), and we can replace D everywhere in the ontology by $\top$.

The rewriting transformations that aim at determining unsatisfiable and tautological concepts are applied on the ontology after each elimination of a symbol, since some of these redundancies are not detected by the redundancy elimination rules. This is again an indispensable step to obtain practicality for larger ontologies.

After all symbols have been eliminated, we use some simple transformations to make the result more human-readable, by shortening axiom sizes and moving negatively occurring concepts to the left-hand side of a concept inclusion.

8.1.3 Role Restriction Resolution

In the methods discussed in Chapters 4, 5 and 7 that are interpolation complete not only for forgetting concept symbols, we do not differentiate between eliminating concept symbols and role symbols. From Theorem 4.4.13 on minimal clausal representations of uniform interpolants, it follows that, in order to eliminate a symbol, one has

Role restriction resolution:

$$\frac{C_0 \sqcup \forall r.D_0 \quad \ldots \quad C_n \sqcup \forall r.D_n \qquad C \sqcup \exists r.D}{C_0 \sqcup \ldots \sqcup C_n \sqcup C} \qquad \mathcal{N} \models D_0 \sqcap \ldots \sqcap D_n \sqcap D \sqsubseteq \bot$$

Figure 8.1: Role restriction resolution rule used in the implementations.

to perform all inferences on that symbol which produce clauses without that symbol. From this perspective, there is no difference in whether we forget a concept symbol or a role symbol. However, from an implementation point of view, forgetting concept symbols is much easier than forgetting role symbols.

The crucial inferences to eliminate a concept symbol are applications of the resolution rule on the symbol to be eliminated. The main challenge when forgetting concept symbols is to determine which inferences on roles have to be performed in order to make these resolution steps possible. On the other hand, the crucial inferences for eliminating role symbols are applications of the monotonicity rules and the $\exists$-elimination rule. In order to determine when the $\exists$-elimination rule can be applied, we have to infer clauses of the form $\neg D$, which can involve inferences on all symbols present in the clause set. This makes a pre-selection of clauses more difficult, as well as finding an appropriate search strategy for eliminating roles. To obtain easier algorithms, we use an additional rule in our implementation. This rule works similar to resolution on concept symbols, which allows us to focus on inferences on the symbol to eliminate. In addition, it makes it possible to use an external reasoner, which further simplifies the implementation.

The rule is the role restriction resolution rule shown in Figure 8.1. It works like a combination of the $\forall\exists$-role propagation rule and the $\exists$-elimination rule, where we leave unspecified how to determine the unsatisfiability of the corresponding introduced definer. In our implementation, we use the OWL reasoner HermiT 1.3.6 (Glimm et al., 2014) to determine which conjunctions of definers are unsatisfiable. Additionally, caching is used to avoid unnecessary requests to the reasoner.

In order to eliminate a role symbol r in an $\mathcal{ALCH}$ ontology or an $\mathcal{ALC}$ knowledge base, we only have to compute all inferences of the monotonicity rules, the role instantiation rules and the role restriction resolution rule that involve r.

8.1.4 Search Strategy 1: Propagate First

Some rules in the calculi are only applied with the aim of making new inferences on a specific symbol possible. This is the case for the role propagation rules of $Int_{\mathcal{ALCH}}$, the role combination rules of $Int_{\mathcal{SHQ}}$, and the role instantiation rules of $Int_{\mathcal{SHI}_\vee}$. In the following, we refer to all these rules as *role combination rules*. Performing all inferences of the role combination rules is not feasible, since this may involve all role symbols present in the ontology. We illustrate this with two examples.

Example 8.1.1. Consider the following set of clauses:

$$\begin{array}{cc} C_1 \sqcup \forall r.D_1 & \neg D_1 \sqcup A_1 \\ & \vdots \\ C_n \sqcup \forall r.D_n & \neg D_n \sqcup A_n \\ C_a \sqcup \forall r.D_a & \neg D_a \sqcup B \\ C_b \sqcup \exists r.D_b & \neg D_b \sqcup \neg B \end{array}$$

Suppose we want to forget B. If we unrestrictedly apply role propagation, we eventually combine every combination of definers possible in this clause set and derive 2^n clauses with role restrictions. However, since we are only interested in inferences on B, most of these applications of the role propagation rules are not necessary. In fact, the only clauses on which we have to apply a role combination rule are $C_a \sqcup \forall r.D_a$ and $C_b \sqcup \exists r.D_b$. The reason is that only the definers D_a and D_b occur together with B in a clause.

Example 8.1.2. Consider the following set of clauses, and suppose again we want to forget B:

$$\begin{array}{cc} C_a \sqcup \forall r.D_{1a} & C_b \sqcup \exists r.D_{1b} \\ \neg D_{1a} \sqcup \forall r.D_{2a} & \neg D_{1b} \sqcup \exists r.D_{2b} \\ & \vdots \\ \neg D_{n-1a} \sqcup \forall r.D_{na} & \neg D_{n-1b} \sqcup \exists r.D_{nb} \\ \neg D_{na} \sqcup B & \neg D_{nb} \sqcup \neg B \end{array}$$

Only D_{na} and D_{nb} occur together with B in a clause, but we cannot directly apply $\forall\exists$-role propagation on $\neg D_{n-1a} \sqcup \forall D_{na}$ and $\neg D_{n-1b} \sqcup \exists r.D_{nb}$, since the resulting clause

would contain more than one negative definer literal. In order to infer a set of clauses on which we can resolve on B, we have to combine each pair of definers D_{ia}, D_{ib} using the $\forall\exists$-role propagation, where $1 \leq i \leq n$, starting with D_{1a} and D_{1b}. If we apply the combination rules unrestrictedly, we perform much more inferences than this.

In order to select which combination rules we have to apply, one simple solution is to select role restrictions based on whether the corresponding definer is connected to the symbol we want to forget by a chain of clauses. This motivates our formal notion of connectedness defined next.

Definition 8.1.3. A definer D is connected to a symbol s if there is a sequence of clauses $\neg D_1 \sqcup C_1 \sqcup \mathsf{Q}_2 R_2.D_2, \neg D_2 \sqcup C_2 \sqcup \mathsf{Q}_3 R_3.D_3, \ldots, \neg D_n \sqcup C$, where $\mathsf{Q}_i \in \{\exists, \forall\}$ for $i \leq n$, $D = D_1$ and $s \in sig(C)$.

In order to forget a symbol s, only role combination rules on definers that are connected to s are necessary. While this can already reduce the number of inferences significantly, this strategy is not sufficient to obtain practicality in most cases. For this, a more refined strategy is required. This is illustrated in Example 8.1.2, in which every definer is connected to B, but not all instances of the role propagation rule have to be applied.

Definition 8.1.4. A definer D has a *distance of* n to a symbol $s \in N_c \cup N_r$, if there is a sequence of n clauses of the form $\neg D_1 \sqcup C_1 \sqcup \mathsf{Q}_2 R_2.D_2$, $\neg D_2 \sqcup C_2 \sqcup \mathsf{Q}_3 R_3.D_3$, …, $\neg D_n \sqcup C_n$, $\mathsf{Q}_i \in \{\exists, \forall\}$ for $i \leq n$, in the current clause set such that $D = D_1$ and $s \in sig(C_n)$.

In the prototype for uniform interpolation of $\mathcal{ALCH}$ ontologies, we only apply role propagation on pairs of definers that either (1) have the same distance to the symbol to be eliminated or (2) are both connected to the symbol to be eliminated and at least one definer is cyclic. Note that a definer is cyclic if and only if it is connected to itself. We obtained a further increase in performance by taking into account the polarity of the symbol with which it is connected to the definer.

8.1.5 Search Strategy 2: Resolve First

For the prototypes for $\mathcal{SHQ}$ ontologies and $\mathcal{ALC}$ knowledge bases, we used a different strategy, which can be implemented more elegantly. The main idea of the first search

strategy is to predict beforehand which role combination inferences are necessary. The second search strategy works the other way around: we start by resolving on the symbol to be eliminated, and determine subsequently which role combination inferences are necessary to make this resolution step possible.

For interpolation completeness in the languages considered in the implementation, it is sufficient to derive clauses with at most one negative definer symbol. The prototype for $\mathcal{ALCH}$ takes this into account by not performing inferences that would produce clauses with more than one negative definer literal. In contrast, the prototypes for $\mathcal{ALC}$ and $\mathcal{SHQ}$ apply resolution and role resolution unrestrictedly. If an inferred clause is of the form $\neg D_1 \sqcup \neg D_2 \sqcup C$, where $D_1, D_2 \in N_d$ and C is possibly empty, we check whether rule applications are possible that trigger the introduction of a definer D_{12} representing $D_1 \sqcap D_2$. If this is the case, we perform this inference, and add the clause $\neg D_{12} \sqcup C$ to the current clause set, together with the clauses necessary for the introduction of this definer. This is done recursively, that means, any clauses derived during this process can trigger the introduction of further definers.

Note that with this search strategy, we apply role combination rules only if they allow us to infer a new resolvent with only one negative definer literal. On the other hand, a drawback with respect to the first strategy is that we may consider resolution steps more often than is necessary, finding out only subsequently that no sequence of role combination rules makes this step possible.

8.2 The Corpus

For the evaluation, we selected 306 ontologies from the NCBO BioPortal repository (Noy et al., 2009). The NCBO BioPortal repository contains ontologies developed for different applications in medicine, biology and bio-informatics. These ontologies are from real-life applications, and differ in size, expressivity and structure. They therefore offer a rich, diverse and realistic test set, which is why we chose it for evaluation of our method. A detailed description of the repository and its ontologies can be found in Horridge et al. (2011) and in Matentzoglu et al. (2013).

The corpus was selected from a snapshot of the repository taken in January 2015, containing 339 ontologies. From this corpus we selected 306 ontologies based on the

following criteria:

1. They contain at most 100,000 axioms.
2. They are satisfiable.
3. They can be processed by the OWL reasoner HermiT 1.3.6.

We did a small investigative test run computing uniform interpolants for signature sizes of 50, with a sample size of 50 signatures per ontology and a timeout of 30 minutes, and found that ontologies with more than 100,000 axioms always caused a timeout or memory-problems, sometimes already in the process of parsing and converting them into our internal representation. For this reason, we did not consider ontologies with this size in our evaluation.

The second criterion is necessary because our current implementation assumes the input ontology to be satisfiable. Satisfiability is not a requirement of our basic algorithms and calculi, which would always compute an unsatisfiable uniform interpolant from an unsatisfiable input ontology. However, the module extractor used in the preprocessing step only computes correct results on satisfiable ontologies (see Section 8.1.2). Therefore, unsatisfiable ontologies had to be removed from the corpus.

The third criterion is a prerequisite to be able to check Criterion 2, which we tested using the OWL reasoner HermiT 1.3.6. The reasoner was used in our implementation for the role restriction resolution rule, as described in Section 8.1.3. For ontologies that can be processed by HermiT, we can also be sure that the conditions on number restrictions discussed in Section 3.1 are fulfilled by all ontologies.

Table 8.1 shows some statistical information about the ontologies in the corpus. For each ontology, the table shows the number of axioms, the average axioms size, and the number of cardinality restrictions occurring in that ontology. The size of an axiom corresponds to the number of occurrences of concept symbols, role symbols, individuals and operators in that axiom. As cardinality restrictions, we counted concepts of the form $\geq nr.C$, $\leq mr.C$ and $=mr.C$ only if $n > 1$ and $m > 0$, since otherwise they are just syntactic variants of existential restrictions, universal restrictions or tautologies. For each value, Table 8.1 shows the mean, the median and 90th percentile values. More detailed information about cardinality restrictions occurring in the repository is given at the beginning of Chapter 6.

Axioms in the Corpus	
Axioms Mean:	4,759.98
Axioms Median:	1,123.00
Axioms 90th percentile:	13,045.00
Average Axiom Size Mean:	3.71
Average Axiom Size Median:	3.53
Average Axiom Size 90th percentile:	4.80
Cardinality Restrictions Mean:	13.16
Cardinality Restrictions Median:	0
Cardinality Restrictions 90th percentile:	7

Table 8.1: Statistics about the corpus used.

For each prototype, we eliminated axioms that are not in the supported language. For example, for the $\mathcal{ALCH}$ prototype, we would remove all ABox axioms and all axioms that use a concept expression that is not in $\mathcal{ALCH}$. In this context, we transformed domain and range restriction axioms as corresponding concept inclusions, and cardinality restrictions of the forms $\geq 0r.C$, $\geq 1r.C$ and $\leq 0r.C$ as corresponding expressions in $\mathcal{ALCH}$. Concepts of the form $=nr.C$ were translating into concepts of the form $\geq nr.C \sqcap \leq nr.C$.

Since the different experiments require the input ontology to have a minimal set of symbols in the signature, we further removed, depending on the experimental setup, ontologies that have a smaller signature than the selected signature size.

8.3 Forgetting Few Symbols

In the first series of experiments, we evaluated the performance of our prototypes for the task of forgetting small numbers of symbols. Forgetting small sets of symbols is the expected task to be performed when computing logical differences of different ontology versions, since from one version to another usually only a small set of symbols will be added or removed. The evaluation therefore gives us an indication of the practicality of uniform interpolation for this application or similar applications where the number of symbols to be eliminated is small.

Since our method processes symbols one after another, this series of experiments can give us an idea about how the number of symbols to be forgotten affects the performance. In this experiment, we did not use a module extractor, to ensure that

$\mathcal{ALCH}$, forget 50 symbols	
Success Rate:	91.10%
Without Fixpoints:	95.29%
Duration Mean:	7.68 sec.
Duration Median:	2.74 sec.
Duration 90th percentile:	12.45 sec.

$\mathcal{ALCH}$, forget 100 symbols	
Success Rate:	88.10%
Without Fixpoints:	93.27%
Duration Mean:	18.03 sec.
Duration Median:	3.81 sec.
Duration 90th percentile:	21.17 sec.

$\mathcal{ALC}$ w. ABoxes, forget 50 symbols	
Success Rate:	94.79%
Without Fixpoints:	92.91%
Duration Mean:	23.94 sec.
Duration Median:	3.01 sec.
Duration 90th percentile:	29.00 sec.

$\mathcal{ALC}$ w. ABoxes, forget 100 symbols	
Success Rate:	91.37%
Fixpoints:	92.48%
Duration Mean:	57.87 sec.
Duration Median:	6.43 sec.
Duration 90th percentile:	99.26 sec.

$\mathcal{SHQ}$, forget 50 concept symbols	
Success Rate:	95.83%
Without Fixpoints:	93.40%
Duration Mean:	7.62 sec.
Duration Median:	1.04 sec.
Duration 90th percentile:	4.89 sec.

$\mathcal{SHQ}$, forget 100 concept symbols	
Timeouts:	90.77%
Fixpoints:	91.99%
Duration Mean:	13.51 sec.
Duration Median:	1.60 sec.
Duration 90th percentile:	11.65 sec.

Table 8.2: Forgetting 50 and 100 symbols with the prototypes.

all symbols are eliminated by the prototypes themselves. The results may therefore indicate how the methods would perform for forgetting larger amounts of symbols in the ontologies.

For each prototype and ontology, we generated random sets of symbols of size 50 and 100, where for each signature size, we generated 360 samples. Since the method for $\mathcal{SHQ}$ does not support forgetting roles, we generated corresponding samples of concept symbols for the $\mathcal{SHQ}$ prototype. For each pair of ontology and symbol set, we eliminated the symbols from the set with a timeout of 30 minutes.

The results are shown in Table 8.2, where for each prototype and signature size, we show the success rate, and for the successful computations, the percentage of uniform interpolants that did not use fixpoint expressions. In addition, the table as well as the mean, median and 90th percentile of the durations of the successful runs. Since in most cases, we only forgot a minority of the symbols of the ontology, the size and the structure of the ontology remained mostly unchanged. For this reason, we did not analyse the axioms of the uniform interpolants computed in this series of experiments.

In around 90–95% percent of cases, the uniform interpolant could be computed

within 30 minutes. In most cases, the uniform interpolants could be computed in just a few seconds, with the median of the duration being below 7 seconds in all cases. Except for the $\mathcal{ALC}$ forgetter with ABox support, the 90th percentile of the duration was always below 22 seconds.

Regarding fixpoint operators, we can see that these were used in a number of cases, but in the majority of cases (always over 90%), the uniform interpolant could be represented without them. Note that, even though we use some syntactic checks to eliminate tautological fixpoint operators, it is still possible that some of the uniform interpolants with fixpoint operators could be equivalently represented without fixpoint operators.

With the $\mathcal{SHQ}$ prototype, we can only forget concept symbols, but not role symbols. It is to be expected that the type of symbols we forget has an effect of the performance of the implementation. Specifically, forgetting role symbols is expected to be a harder task than forgetting concept symbols. This can be explained by the following reasons. (1) Most ontologies contain much fewer role symbols than concept symbols, and use each role symbol in a larger amount of axioms. Forgetting a single role symbol therefore involves the processing of more axioms than forgetting a single concept symbol. (2) Whereas the standard resolution rule only applies to a pair of clauses, the role restriction resolution rule applies to an arbitrary set of clauses. (3) The role restriction resolution rule makes use of an external reasoner, which makes inferences by this rule more expensive operations than inferences by the normal resolution rule.

To get a clear understanding of these effects, we performed another sequence of experiments for the $\mathcal{ALCH}$ and $\mathcal{ALC}$ prototypes, where we either only eliminated concept symbols or only eliminated role symbols. Again, we generated 360 signatures for each experiment and set the timeout to 30 minutes. For concept symbols, we selected 50 symbols in each case, and for role symbols 5. The results are shown in Table 8.3.

The table supports our hypothesis that a timeout is more likely to happen if role symbols are eliminated. When eliminating only concept symbols, we had a success rate of 96.05% for the $\mathcal{ALC}$ forgetter with ABox support, and a success rate of 98.12% for the $\mathcal{ALCH}$ forgetter. These values are much higher than the corresponding values

$\mathcal{ALCH}$, forget 50 concepts	
Success Rate:	98.12%
Without Fixpoints:	90.60%
Duration Mean:	8.53 sec.
Duration Median:	2.69 sec.
Duration 90th percentile:	12.60 sec.

$\mathcal{ALCH}$, forget 5 roles	
Success Rate:	86.20%
Without Fixpoints:	100.00%
Duration Mean:	6.69 sec.
Duration Median:	0.72 sec.
Duration 90th percentile:	5.38 sec.

$\mathcal{ALC}$ w. ABoxes, forget 50 concepts	
Success Rate:	96.05%
Without Fixpoints:	91.75%
Duration Mean:	26.70 sec.
Duration Median:	2.93 sec.
Duration 90th percentile:	28.14 sec.

$\mathcal{ALC}$ w. ABoxes, forget 5 roles	
Success Rate:	91.17%
Without Fixpoints:	100.00%
Duration Mean:	17.63 sec.
Duration Median:	0.54 sec.
Duration 90th percentile:	6.55 sec.

Table 8.3: Forgetting concept and role symbols exclusively.

in the first experiment, where we forgot both concept and role symbols. In contrast, timeouts occurred more often than in the first experiment when we only eliminated roles, even though the number of forgotten symbols was much lower. This shows that forgetting role symbols is much harder to perform for our prototypes than forgetting concept symbols.

Note that the results of forgetting role symbols never contained fixpoint expressions. Even though one can construct ontologies for which the result of forgetting a role symbol requires a fixpoint operator, in practice this is unlikely to happen.

8.4 Uniform Interpolants for Small Signatures

In this section, we discuss the results of a series of experiments to compute uniform interpolants for small signatures. These uniform interpolants are particularly useful for analysing ontologies, as we argue in Section 1.1.

Again we generated for each ontology 360 random signatures of size 50 and 360 random signatures of size 100. This time, we used module extraction as a preprocessing step, to reduce the number of axioms to be processed (see Section 8.1.2). In the case of the prototype for $\mathcal{SHQ}$, we only eliminated concept symbols from the extracted modules. With the other prototypes, we eliminated all symbols that were not in the selected signature, such that the resulting ontology only contained symbols in this signature. The timeout was again set to 30 minutes.

$\mathcal{ALCH}$ Ontologies, $\#\mathcal{S} = 50$	
Success Rate:	78.64%
Without Fixpoints:	98.30%
Duration Mean:	152.61 sec.
Duration Median:	56.89 sec.
Duration 90th percentile:	427.53 sec.
Axioms Mean:	80.88
Axioms Median:	21.00
Axioms 90th percentile:	177.00
Ax. Size Mean:	6.43
Ax. Size Median:	3.00
Ax. Size 90th percentile:	5.41

$\mathcal{ALCH}$ Ontologies, $\#\mathcal{S} = 100$	
Success Rate:	76.87%
Without Fixpoints:	97.75%
Duration Mean:	175.00 sec.
Duration Median:	73.21 sec.
Duration 90th percentile:	474.53 sec.
Axioms Mean:	210.08
Axioms Median:	66.00
Axioms 90th percentile:	320.00
Ax. Size Mean:	5.13
Ax. Size Median:	3.05
Ax. Size 90th percentile:	5.38

$\mathcal{ALC}$ Knowledge Bases, $\#\mathcal{S} = 50$	
Success Rate:	84.78%
Without Fixpoints:	96.06%
Duration Mean:	113.90 sec.
Duration Median:	29.58 sec.
Duration 90th percentile:	330.56 sec.
Axioms Mean:	198.52
Axioms Median:	31.00
Axioms 90th percentile:	426.00
Ax. Size Mean:	6.15
Ax. Size Median:	3.00
Ax. Size 90th percentile:	5.59

$\mathcal{ALC}$ Knowledge Bases, $\#\mathcal{S} = 100$	
Success Rate:	80.54%
Without Fixpoints:	95.04%
Duration Mean:	313.28 sec.
Duration Median:	214.56 sec.
Duration 90th percentile:	780.30 sec.
Axioms Mean:	302.78
Axioms Median:	84.00
Axioms 90th percentile:	709.00
Ax. Size Mean:	4.66
Ax. Size Median:	3.04
Ax. Size 90th percentile:	5.82

$\mathcal{SHQ}$ Ontologies, $\#(\mathcal{S} \setminus N_r) = 50$	
Success Rate:	77.63%
Without Fixpoint:	91.15%
Duration Mean:	256.56 sec.
Duration Median:	184.66 sec.
Duration 90th percentile:	551.90 sec.
Axioms Mean:	143.40
Axioms Median:	49.00
Axioms 90th percentile:	302.00
Axiom Size Mean:	201.91
Ax. Size Median:	4.50
Ax. Size 90th percentile:	19.69
Card. Mean:	431.88
Card. Median:	0.00
Card. 90th percentile:	8.00

$\mathcal{SHQ}$ Ontologies, $\#(\mathcal{S} \setminus N_r) = 100$	
Success Rate:	69.35%
Without Fixpoints:	93.50%
Duration Mean:	380.56 sec.
Duration Median:	269.03 sec.
Duration 90th percentile:	975.74 sec.
Axioms Mean:	303.22
Axioms Median:	100.00
Axioms 90th percentile:	536.00
Ax. Size Mean:	23.61
Ax. Size Median:	4.12
Ax. Size 90th percentile:	16.12
Card. Mean:	245.80
Card. Median:	0.00
Card. 90th percentile:	11.00

Table 8.4: Results of computing uniform interpolants for small signature sizes.

The results of these experiments are shown in Table 8.4. In contrast to the experiments presented in the last section, in the current setup most uniform interpolants had a completely different syntactical structure than the corresponding input ontologies. For this reason, we did not only protocol the number of timeouts and uniform interpolants with fixpoints, and the duration of each run, but also details about the form of the computed uniform interpolants.

Table 8.4 shows for each prototype and signature size the percentage of uniform interpolants that could be computed within the timeout, and of the successfully computed uniform interpolants, the proportion of ontologies that contained fixpoint expressions, and statistical values about the duration of the computation, the number of axioms in the uniform interpolant and the average axiom size in the uniform interpolant. For the $\mathcal{SHQ}$ uniform interpolants, we additionally show statistical values about the number of cardinality restrictions in the uniform interpolants. For each value, we present the mean, the median and the 90th percentile.

As could be expected, the success rate is lower than for the experiments presented in the last section. This is partly caused by the larger number of symbols to be eliminated, especially in very large ontologies, since this required a larger number of inferences to be performed. For between 36 and 41 ontologies, a uniform interpolant could never be computed in the specified time frame, since the ontologies were simply too large. Even though we used module extraction as a preprocessing step, the resulting ontology often still contained a large number of axioms and symbols to be eliminated.

Another reason for the high number of timeouts can be explained by the selection of signatures. We found that the distribution of symbols in an ontology is highly unequal, with few symbols occurring very frequently and the majority of symbols occurring only rarely. This becomes evident in Figure 8.2, where we plotted the distribution of symbol frequencies for all ontologies in the NCBO BioPortal repository.

Each ontology is represented by a red line, and each point in the line represents a single symbol in the signature of this ontology. On the y-axis, we plot the number of axioms in which that symbol occurs, and the values are sorted along the x-axis. The average values for all lines are shown in blue. Both the x- and the y-axis are in logarithmic scale. We can see that a few number of symbols occurs in a lot of axioms, whereas the majority occurs only in a very small number. For example, on average,

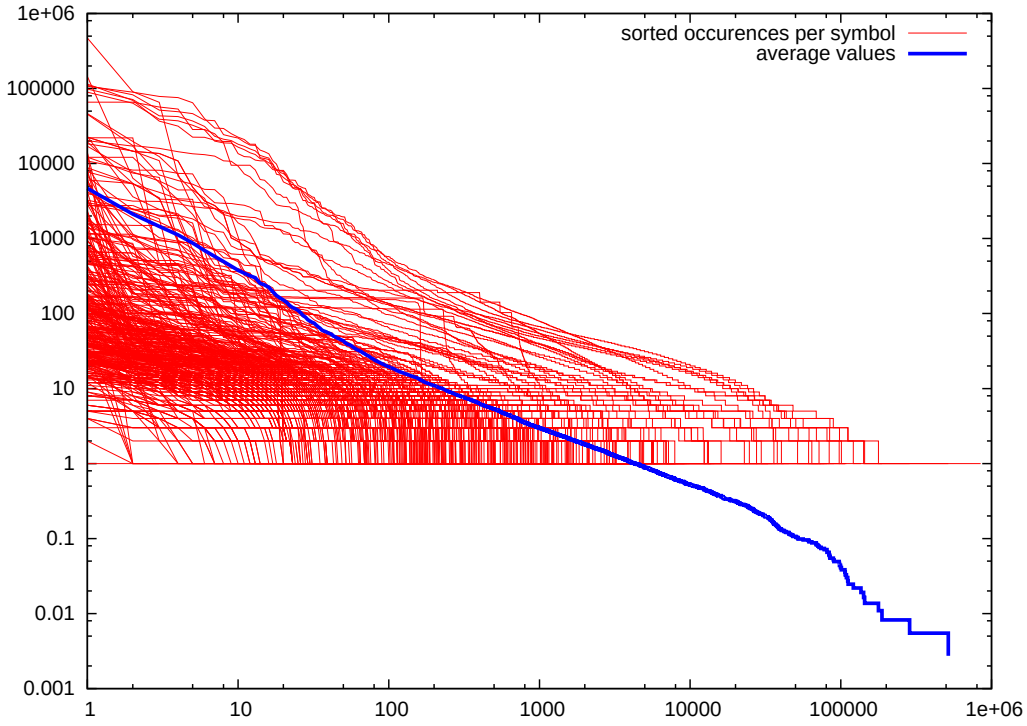

Figure 8.2: Logarithmic distribution of symbol frequencies in ontologies of the NCBO BioPortal repository.

all but 100 symbols occur less than 12 axioms.

Since we eliminated the majority of symbols from the ontology in most cases, the probability that we had to eliminate a symbol occurring more often than average was very high. The effect of this will be discussed more deeply in Section 8.5.

For the uniform interpolants that could be computed in the specified time, we find that their structure was usually well-suited for the targeted application of ontology analysis. Fixpoints were only used in rare cases, which can be explained by the fact that we eliminated most of the role symbols in most cases. The median of the average axiom size in an ontology was between 3.00 and 4.50, which implies the majority of axioms were simple concept inclusions of the form $A \sqsubseteq B$. The median for the number of axioms was 21 and 100, which are reasonable numbers for signatures of sizes 50 and 100. Note that a small signature also restricts the number of different axioms that can occur in the uniform interpolant, unless the axioms are very complex. This is especially the case if the signature contained no role symbols. The situation was a little different for knowledge bases, since the number of individuals in the uniform interpolant is the same as in the original knowledge base. This explains the higher axiom numbers in the 90th percentile for the $\mathcal{ALC}$ forgetter with support for ABox axioms.

Whereas the median values for the average axiom size in $\mathcal{SHQ}$ uniform interpolants were comparable to the respective values for the other prototypes, we can see that the corresponding mean values are highly distorted by outliers with large axiom sizes. For instance, for a signature size of 100, the mean of the average axiom size is 201.91, whereas the 90th percentile was only 16.69. Therefore, a small number of uniform interpolants computed by the $\mathcal{SHQ}$ prototype were very complex. This can be partly explained the complexity of the rules used by the $\mathcal{SHQ}$ calculus, which enable us to infer more complex axioms and larger axioms than for the other description logics. This is necessary due to the higher expressivity of $\mathcal{SHQ}$, which allows us to preserve more entailments in the uniform interpolant than $\mathcal{ALC}$ and $\mathcal{ALCH}$. This was also reflected by the high average number of number restrictions in some of the uniform interpolants, as the mean of this variable indicates.

The main reason however for the higher complexity of axioms in the $\mathcal{SHQ}$ uniform interpolants is that we did not eliminate roles symbols. The more complex axioms

in the $\mathcal{SHQ}$ uniform interpolants contained deep nestings of role restrictions, since a lot of the structure of the corresponding module had to be preserved in the uniform interpolant. In contrast, for the other description logics, the majority of the roles was typically eliminated, which restricts the information to be preserved in the resulting axioms. This is also the reason why the number of uniform interpolants with fixpoint expressions was very low for $\mathcal{ALCH}$ and $\mathcal{ALC}$, whereas for $\mathcal{SHQ}$ it was comparable with the results presented in the last section.

8.5 Uniform Interpolants for Central Signatures

Whereas in the previous experimental set-ups, we selected symbols randomly, for this section we evaluated the performance of the prototypes for signatures that can be regarded as central to the ontology. An application of this is the computation of ontology summaries that was discussed in Section 1.1. Computing uniform interpolants for the central symbols in an ontology is also interesting from another point of view. As it turns out, some symbols are harder to eliminate from an ontology than other symbols. If a symbol is used frequently in an ontology, it is likely that it is harder to eliminate. On the other hand, for applications like ontology reuse, information hiding and ontology analysis, it may not always be necessary to eliminate the symbols that play a central role in the ontology. If computing uniform interpolants for central symbols turns out to be easier than for random signatures, this would support the usefulness of uniform interpolants for these applications.

In general, the central symbols in an ontology can only be known by the ontology designer or an expert in the field. However, to be able to automate the experiments, we used two simple heuristics for the selection of symbols.

The first heuristic is based on frequency of occurrences. One can reasonably expect that symbols that are used most often should play a central role in the ontology. For example, in an ontology about the partonomy of the human body, the role “hasPart” will play a central role, and it will be used in a lot of axioms.

An advantage of this heuristic is that the motivation is intuitive and that it can very easily be computed. A disadvantage is that it is a purely syntactical heuristic and might overlook semantic relations between symbols. To illustrate this, suppose the

ontology about the partonomy of the human body also contains the role "isPartOf", which is defined as the inverse of "hasPart". Since both roles express the same relation, one could argue that they have an equal importance in the ontology, even if "hasPart" is used in the majority of axioms and "isPartOf" only in one axiom. Also, in the result of forgetting "hasPart" from this ontology, "isPartOf" would likely occur as frequently as "hasPart" in the original ontology.

The second heuristic uses genuine modules to determine the "influence" of a symbol. Genuine modules have been developed as a means to examine the modular structure of an ontology (Del Vescovo et al., 2011). A genuine module, as it is defined in Del Vescovo et al. (2011), is the $\top\bot*$-module for the signature of a single axiom in the ontology. For a given axiom α, the corresponding genuine module is the smallest $\top\bot*$-module that contains α.

For the second heuristic, we measure the importance of a symbol based on the number of genuine modules of an ontology that contain it. The intuition is the following. If a symbol s occurs in n genuine modules, then there are n axioms that usually have to be included in modules that preserve entailments in signatures containing s. One can argue that this reflects how the meaning of s is "influenced" by these axioms, or conversely the meaning of how many concepts defined in the ontology depends on interpretations of s. This suggests that s is more central to an ontology if it occurs in a lot of genuine modules.

Note that this measure is independent of the frequency of s. Consider an ontology about family relations that contains a concept "Person" which has a lot of subconcepts. For every axiom that contains a subconcept of "Person" positively, the corresponding genuine module would contain "Person" as well, and therefore "Person" would get assigned a high value even if it itself only occurs in a small number of axioms. Similarly, for the example given above, "hasPart" and "isPartOf" would get the same value assigned, even if the number of occurrences is very different. As it turns out, uniform interpolants of symbols that occur in the most genuine modules can be more cheaply computed than uniform interpolants for signatures that are based on frequency of occurrence.

The number of genuine modules in which symbols occurred in the repository is plotted in Figure 8.3. The graph shows the values for all but 39 ontologies of the

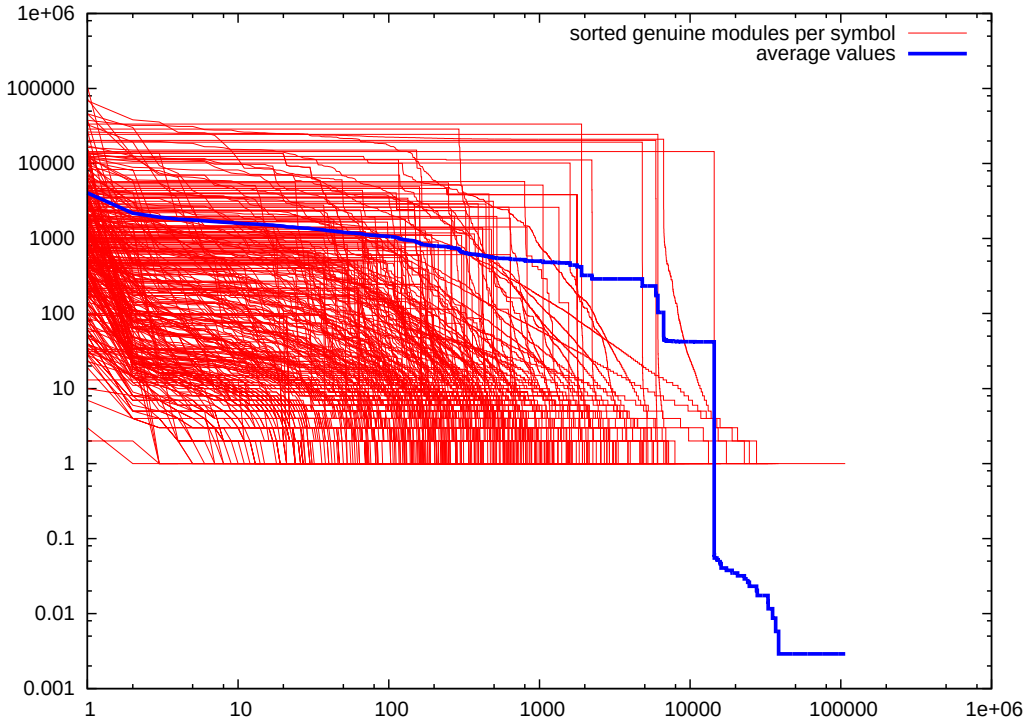

Figure 8.3: Logarithmic distribution genuine modules per symbol in ontologies of the NCBO BioPortal repository.

Heuristic based on frequency	
Success Rate:	85.60 %
Without Fixpoints:	93.15%
Duration Mean:	25.63 sec.
Duration Median:	1.45 sec.
Duration 90th percentile:	11.14 sec.
Axioms Mean:	122.67
Axioms Median:	63.50
Axioms 90th percentile:	202.00
Ax. Size Mean:	10.79
Ax. Size Median:	4.12
Ax. Size 90th percentile:	9.16

Heuristic based on genuine modules	
Success Rate:	92.60%
Without Fixpoints:	97.81%
Duration Mean:	6.34 sec.
Duration Median:	0.61 sec.
Duration 90th percentile:	1.35 sec.
Axioms Mean:	65.66
Axioms Median:	52.50
Axioms 90th percentile:	112.00
Ax. Size Mean:	4.20
Ax. Size Median:	4.00
Ax. Size 90th percentile:	5.16

Table 8.5: Results for computing $\mathcal{ALCH}$ uniform interpolants for signatures of size 50, selected based on two different heuristics.

repository. For the remaining ontologies, the computation of all genuine modules either took too long or caused a memory error. As for Figure 8.2, in which we plotted frequencies of concept symbols, the distribution for each individual ontology is visualised by a red line. Each line represents a single ontology, and each point in that line represents a single symbol in that ontology. The y-axis shows the number of genuine modules in which the symbol occurred, and we sorted the values along the x-axis. The blue line shows the average values for all lines. Both the x-axis and the y-axis are in logarithmic scale. The graph has many steps, since a lot of symbols occurred together in the same set of genuine modules. Even though there are large steps in the graph, the values between the steps are very different. We can see that the distribution for this heuristic is very unequally distributed as well, even though the curves are not as smoothly falling as for the frequencies of occurrences.

Based on these two heuristics, we computed $\mathcal{ALCH}$ uniform interpolants for signatures of size 50. In the first experiment, we selected for each ontology the 50 symbols that occurred in the highest number of axioms . In the second experiment, we selected for each ontology the 50 symbols that occurred in the highest number of genuine modules. The results are shown in Table 8.5.

For both heuristics, the success rate was higher than for random signatures, and uniform interpolants could be computed significantly faster. The best results were however obtained for symbols selected based on the number of genuine modules in which they occur. In comparison, when we selected signatures of size 50 randomly,

the success rate was at 78.64%, and the average duration was 152.61 seconds. When we selected signatures of 50 symbols based on the number of genuine modules in which they occur, the success rate was 92.60%, and the average duration was 6.34 seconds. Moreover, 90% of the uniform interpolants could be computed in less or equal than 1.35 seconds.

In general, the average axiom size was larger than for random signatures. This can be explained by the fact that the signatures we selected using heuristics are more likely to contain roles, which made it possible to express more complex axioms. This suggests that the axioms contained more substantial information. Note that the size was still not too large to affect human readability of the axioms, with few exceptions in the outliers.

The results indicate that for practical applications, it might be possible to compute uniform interpolants in even more cases than the results in the previous sections indicate, if the corresponding signature contains symbols which occur frequently or occur in a lot of genuine modules.

Chapter 9

Conclusion and Future Directions

In this thesis, we investigated saturation-based procedures for the computation of uniform interpolation in various expressive description logics. Uniform interpolation has potential applications in a lot of different areas such as ontology analysis, ontology evolution and information hiding. Since uniform interpolation in expressive description logics is a potentially difficult problem, methods for the construction of uniform interpolants have to proceed in a goal-oriented manner. For this reason, we developed a range of new resolution-based calculi, which enable us to compute inferences on specific concept and role symbols. Whereas most calculi are optimised towards a specific reasoning task, such as deciding satisfiability or computing classification trees, we introduced the new notion of interpolation completeness, which captures the suitability of a calculus for the computation of uniform interpolants. Using an interpolation complete calculus, it is possible to compute all inferences on the symbols to be eliminated, such that all occurrences of these symbols can be removed for the uniform interpolant.

Because most classical description logics do not have uniform interpolation, that is, finite uniform interpolants do not always exist in these languages, we considered for each logic its extension with fixpoint operators. Using a flattened normal form, combined with a dynamic introduction of new symbols, the calculi always compute finite saturations, even if the corresponding uniform interpolant cannot be finitely represented without fixpoints. Interpolation completeness of the calculi makes sure that all symbols that have been introduced can be eliminated using simple rewrite rules, possibly introducing fixpoint expressions.

Language	Refutation Calculus	Interpolation Calculus
$\mathcal{ALC}$	Resolution, $\forall\exists$-role propagation, $\exists$-elimination	$Res_{\mathcal{ALC}}$ + $\forall\forall$-role propagation
$\mathcal{ALCH}$	$Res_{\mathcal{ALC}}$ + $\exists$-monotonicity; variation of role propagation rules if used with redundancy elimination	$Res_{\mathcal{ALCH}}$ + $Int_{\mathcal{ALC}}$ + $\forall$-monotonicity, role hierarchy
$\mathcal{SH}$	$Res_{\mathcal{ALCH}}$ + transitivity rule	$Res_{\mathcal{SH}}$ + $Int_{\mathcal{ALCH}}$
$\mathcal{ALCHI}$	$Res_{\mathcal{ALCH}}$ + role inversion rule	$Res_{\mathcal{ALCHI}}$ + $Int_{\mathcal{ALCH}}$
$\mathcal{ALCHF}$	$Res_{\mathcal{ALCH}}$ + $\forall\forall$- and $\exists\exists$-role propagation, $\geq$2-elimination I and II, $\leq$1- and $\geq$2-monotonicity	$Res_{\mathcal{ALCHF}}$
$\mathcal{SIF}$	$Res_{\mathcal{SH}}$ + $Res_{\mathcal{ALCHI}}$ + $Res_{\mathcal{ALCHF}}$ + universalisation; several reasoning stages	$Res_{\mathcal{SIF}}$
$\mathcal{SHQ}$	Resolution, $\geq$-combination, $\geq\leq$-combination	$Res_{\mathcal{SHQ}}$ + $\leq$-combination, $\leq\geq$-combination, $\geq$-resolution, $\geq$-elimination
$\mathcal{SHI}_\vee$ (ABoxes)	$Res_{\mathcal{SH}}$ + $Res_{\mathcal{ALCHI}}$ + role instantiation, role assertion monotonicity; uses unification and several reasoning stages	$Res_{\mathcal{SHI}}$ + $Int_{\mathcal{ALCH}}$

Table 9.1: Overview of the calculi developed in the thesis.

The introduced symbols furthermore provide a simple way to approximate the uniform interpolant, and can serve as auxiliary concepts to allow for a finite representation of the result without fixpoints, captured in the notion of uniform interpolant modulo direct cycles (Definition 4.1.6 in Chapter 4). This way, the uniform interpolants can be represented as OWL ontologies, and be processed by a state-of-the-art description logic reasoners.

For each description logic $\mathcal{L}$ considered, we developed a refutationally complete calculus $Res_{\mathcal{L}}$ and extended it to an interpolation complete calculus $Int_{\mathcal{L}}$. Table 9.1 gives an overview of these calculi. Three of these methods have been implemented and evaluated, showing practicality in a lot of use cases.

To summarise, the contributions of the thesis are the following:

- The first method for uniform interpolation in $\mathcal{ALC}$ which always terminates with a finite representation of the uniform interpolant, which is also the first practical method that is able to eliminate role symbols.

- Methods for uniform interpolation in 6 additional description logics, which extend $\mathcal{ALC}$ up to the expressivities of $\mathcal{SH}$, $\mathcal{SIF}$ and $\mathcal{SHQ}$, and for which uniform

interpolation has not been investigated before.

- A method for uniform interpolation of $\mathcal{SHI}$ knowledge bases. This is also the first method for uniform interpolation with ABoxes in an expressive description logic that does not have limitations on the structure of the ABox.

- A family of new saturation-based reasoning methods for all logics considered, which uses a new technique of introducing symbols automatically.

- An evaluation showing practicality of the approaches for a lot of cases

- Two simple heuristics for selecting signatures for which uniform interpolants are easy to compute.

The research presented in this thesis does not only have a great potential for a variety of applications that can benefit from using practical uniform interpolation methods. It can also serve as basis for a variety of different future research topics.

9.1 Future Directions

Uniform Interpolation for More Expressive Description Logics. The most expressive description logics we considered in this thesis are $\mathcal{SHI}$, $\mathcal{SIF}$ and $\mathcal{SHQ}$. A reasonable next step would be to consider unions of these description logics, that is $\mathcal{SHIF}$ and $\mathcal{SHIQ}$. As indicated at the end of Chapter 5, this might not be possible unless a more expressive description logic is used for the result. In order to support the full standard OWL DL 2.0, we would further have to investigate uniform interpolation methods for languages with nominals, complex role inclusions, the "self"-operator expressing local reflexivity, and data types. We think that nominals are likely to be the most problematic construct here, due to undecidability results for $\mathcal{ALCO}$ and $\mathcal{ALCHOIQ}$ for the related problems module extraction and deciding conservative extensions (Sattler et al., 2009; Lutz et al., 2007). On the other hand, as shown in Benedikt et al. (2015), unary negation fixpoint logic, which is more expressive than $\mathcal{ALCHOIQ}$, has uniform interpolation, which shows that uniform interpolants can be represented in extensions of description logics with nominals.

To give an idea of how methods for ontologies can be extended to support ABoxes, we developed a method for $\mathcal{SHI}$. An open problem is how to integrate same-as axioms for individuals and number restrictions into the approach. We already solved the latter for ontologies. It is possible that using a similar approach, this method can be extended to support ABoxes, possibly using ideas from the superposition calculus, a method for resolution-based reasoning on clauses with equations (Bachmair and Ganzinger, 1994, 2001).

Role Forgetting in More Expressive Description Logics. If a description logic supports role hierarchies and either transitive roles, functional role restrictions or number restrictions, our methods can only be used to eliminate non-transitive roles, or can eliminate no role symbols at all. Our examples indicate that for these description logics, the target language requires more expressivity to represent the result of forgetting role symbols finitely. Extensions to these results may require the use of Boolean role constructors and complex role inclusion axioms in the target language. It is therefore worth investigating uniform interpolation methods for description logics that support these constructs.

Implementation. Our evaluation shows that our prototypical implementations achieve good results in a lot of realistic use cases. These were however solely implemented as prototypes to show the general practicality of the developed methods. For realistic applications, there is potential of gaining better results by using a more optimised implementation. This could unfold the full potential of uniform interpolation, especially in applications dealing with larger ontologies, or very large ABoxes. For example, efficiency could be improved by using different data structures and multi-threading for the different symbols to be eliminated. Moreover, results from the areas of first-order theorem proving and consequence-based reasoning for description logics could provide another source for more efficient implementations.

Extracting Small Modules. Modules extracted using current approaches often contain more symbols than required, whereas uniform interpolants only use required symbols, but may involve more complex axioms in some cases. A tight integration of both approaches could result in a method for extracting much smaller ontologies for reuse than existing methods. This could for example be achieved by a dynamic adaptation of the signature for which the uniform interpolant is computed. Further

potential lays in using techniques that optimise succinctness of ontologies (Nikitina and Schewe, 2013a,b).

Further Applications of the Developed Calculi. A variety of saturation-based reasoning methods has been developed as part of the project. Apart from uniform interpolation, these could be useful for a variety of other applications. It is worth investigating how the calculi developed in this thesis can contribute to reasoning methods for classification, abduction or approximation. Classification has been a major application of saturation-based reasoning methods that have been developed in the last years. We believe that saturation-based reasoning has another potential application in abduction and approximation. The methods presented in this thesis can already be used for TBox abduction and approximation based on signatures (see Section 1.1, and also Koopmann and Schmidt (2014b, 2015c)). An open problem is how the calculi can be extended for ABox abduction. Another interesting question is whether saturation-based reasoning methods can be used to approximate ontologies into less expressive description logics, a topic that has received a lot of interest in the last years (see for example Pan and Thomas, 2007; Botoeva et al., 2010; Ren et al., 2010; Lutz et al., 2012; Carral et al., 2014).

Bibliography

Wilhelm Ackermann. Untersuchungen über das Eliminationsproblem der mathematischen Logik. *Mathematische Annalen*, 110(1):390–413, 1935.

Carlos Areces and Daniel Gorín. Resolution with order and selection for hybrid logics. *Journal of Automated Reasoning*, 46(1):1–42, 2011.

Carlos Areces, Maarten De Rijke, and Hans De Nivelle. Resolution in modal, description and hybrid logic. *Journal of Logic and Computation*, 11(5):717–736, 2001a.

Carlos Areces, Rosella Gennari, Juan Heguiabehere, and Maarten De Rijke. Tree-based heuristics in modal theorem proving. In *Proceedings of the 13th Belgian-Netherlands Conference on Artificial Intelligence (BNAIC 2001)*. Citeseer, 2001b.

Carlos Areces, Patrick Blackburn, Bernadette Martínez Hernández, and Maarten Marx. Handling Boolean ABoxes. In *Proceedings of the 16th International Workshop of Description Logics (DL 2003*, volume 81 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2003.

Yves Auffray, Patrice Enjalbert, and Jean-Jacques Hebrard. Strategies for modal resolution: results and problems. *Journal of Automated Reasoning*, 6(1):1–38, 1990.

Franz Baader and Werner Nutt. Basic description logics. In Franz Baader, Diego Calvanese, Deborah L. McGuiness, Daniele Nardi, and Peter F. Patel-Schneider, editors, *The Description Logic Handbook*, chapter 2. Cambridge University Press, second edition, 2007.

Franz Baader and Ulrike Sattler. An overview of tableau algorithms for description logics. *Studia Logica*, 69(1):5–40, 2001.

Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, 2005.

Matthias Baaz, Uwe Egly, and Alexander Leitsch. Normal Form Transformations. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 5, pages 273–334. Elsevier, 2001.

Leo Bachmair and Harald Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994.

Leo Bachmair and Harald Ganzinger. Resolution Theorem Proving. In *Handbook of Automated Reasoning*, pages 19–99. Elsevier and MIT Press, 2001.

Leo Bachmair, Harald Ganzinger, and Uwe Waldmann. Refutational theorem proving for hierarchic first-order theories. *Applicable Algebra in Engineering, Communication and Computing*, 5(3):193–212, 1994.

Kenneth Baclawski, Tudor Groza, Torsten Hahmann, and Ivan Varzinczak. *Proceedings of the 8th International Workshop on Modular Ontologies (WoMO 2014)*. CEUR-WS.org, 2014.

Andrew Bate, Boris Motik, Bernardo Cuenca Grau, František Simančík, and Ian Horrocks. Extending consequence-based reasoning to $\mathcal{SHIQ}$. In *Proceedings of the 28th International Workshop on Description Logics*, volume 1350 of *CEUR Workshop Proceedings*, pages 34–46. CEUR-WS.org, 2015.

Michael Benedikt, Balder ten Cate, and Michael Vanden Boom. Interpolation with decidable fixpoint logics. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 378–389. IEEE, 2015.

Meghyn Bienvenu. Complexity of abduction in the $\mathcal{EL}$ family of lightweight description logics. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference (KR-08)*, pages 220–230. AAAI Press, 2008.

Marta Bílková. Uniform Interpolation and Propositional Quantifiers in Modal Logics. *Studia Logica*, 85(1):1–31, 2007.

George Boole. *An investigation of the laws of thought: on which are founded the mathematical theories of logic and probabilities*. Dover Publications, 1854.

Elena Botoeva, Diego Calvanese, and Mariano Rodriguez-Muro. Expressive approximations in DL-Lite ontologies. In *Artificial Intelligence: Methodology, Systems, and Applications*, pages 21–31. Springer, 2010.

Elena Botoeva, Roman Kontchakov, Vladislav Ryzhikov, Frank Wolter, and Michael Zakharyaschev. Query inseparability for description logic knowledge bases. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR 2014)*. AAAI Press, 2014.

Diego Calvanese and Giuseppe De Giacomo. Expressive description logics. In *The Description Logic Handbook*, chapter 5, pages 193–236. Cambridge University Press, 2007.

Diego Calvanese, Giuseppe De Giacomo, and Riccardo Rosati. A note on encoding inverse roles and functional restrictions in $\mathcal{ALC}$ knowledge bases. *Proceedings of the 13th International Workshop on Description Logics (DL 1998)*, 1998.

Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Reasoning in expressive description logics with fixpoints based on automata on infinite trees. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99*, pages 84–89. Morgan Kaufmann, 1999.

David Carral, Cristina Feier, Bernardo Cuenca Grau, Pascal Hitzler, and Ian Horrocks. $\mathcal{EL}$-ifying ontologies. In *Automated Reasoning*, volume 8562 of *Lecture Notes in Computer Science*, pages 464–479. Springer, 2014.

Christian Collberg, Clark Thomborson, and Douglas Low. Manufacturing cheap, resilient, and stealthy opaque constructs. In *Proceedings of the 25th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 184–196. ACM, 1998.

Willem Conradie, Valentin Goranko, and Dimiter Vakarelov. Algorithmic correspondence and completeness in modal logic. I. the core algorithm SQEMA. *Logical Methods in Computer Science*, 2006.

William Craig. Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. *The Journal of Symbolic Logic*, 22(3):269–285, 1957.

Giovanna D'Agostino and Marco Hollenberg. Uniform interpolation, automata and the modal μ-calculus. *Logic Group Preprint Series*, 165, 1996.

Giovanna D'Agostino and Giacomo Lenzi. On modal μ-calculus with explicit interpolants. *Journal of Applied Logic*, 4(3):256–278, 2006.

Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201–215, July 1960.

Chiara Del Vescovo, Bijan Parsia, Ulrike Sattler, and Thomas Schneider. The modular structure of an ontology: Atomic decomposition. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI11)*, pages 2232–2237. AAAI Press, 2011.

Patrick Doherty, Witold Łukaszewicz, and Andrzej Szałas. Computing circumscription revisited: A reduction algorithm. *Journal of Automated Reasoning*, 18(3):297–336, 1997.

Patrick Doherty, Witold Lukaszewicz, and Andrzej Szalas. Computing strongest necessary and weakest sufficient conditions of first-order formulas. In *AI 2001: Advances in Artificial Intelligence*, volume 2256 of *Lecture Notes in Computer Science*, pages 145–154. Springer, 2001.

Patrick Doherty, Andrzej Szalas, and Witold Lukaszewicz. Approximative query techniques for agents with heterogeneous ontologies and perceptive capabilities. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004)*, pages 459–468. AAAI Press, 2004.

Corinna Elsenbroich, Oliver Kutz, and Ulrike Sattler. A case for abductive reasoning over ontologies. In *OWL: Experiences and Directions (OWLED 2006)*, volume 216 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.

Thorsten Engel. Quantifier elimination in second-order predicate logic. Master's thesis, Max-Planck-Institut für Informatik, 1996.

Patrice Enjalbert and Luis Fariñas del Cerro. Modal resolution in clausal form. *Theoretical Computer Science*, 65(1):1–33, 1989.

Luis Fariñas del Cerro. A simple deduction method for modal logic. *Information Processing Letters*, 14(2):49–51, 1982.

Luis Fariñas del Cerro. Resolution modal logic. *Logique et Analyse*, 28(110-111): 153–172, 1985.

Nasim Farsiniamarj and Volker Haarslev. Practical reasoning with qualified number restrictions: A hybrid ABox calculus for the description logic $\mathcal{SHQ}$. *AI Communications*, 23(2):205–240, 2010.

Melvin Fitting. Destructive modal resolution. *Journal of Logic and Computation*, 1 (1):83–97, 1990.

Dov Gabbay and Hans Jürgen Ohlbach. Quantifier elimination in second-order predicate logic. In *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR'92)*, pages 425–435. Morgan Kaufmann, 1992.

Dov M. Gabbay, Renate A. Schmidt, and Andrzej Szałas. *Second Order Quantifier Elimination: Foundations, Computational Aspects and Applications*, volume 12 of *Studies in Logic*. College Publications, 2008.

Peter Gärdenfors. *Knowledge in Flux—Modelling the Dynamics of Epistemic States*. MIT Press, Cambridge, 1988.

William Gatens, Boris Konev, and Frank Wolter. Lower and upper approximations for depleting modules of description logic ontologies. In *ECAI 2014 - 21st European Conference on Artificial Intelligence*. IOS Press, 2014.

Gene Ontology Consortium. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Research*, 32:258–261, 2004.

Silvio Ghilardi and Marek Zawadowski. Undefinability of propositional quantifiers in the modal system S4. *Studia Logica*, 55(2):259–271, 1995.

Silvio Ghilardi, Carsten Lutz, and Frank Wolter. Did I damage my ontology? *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 187–197, 2006.

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.

Bernardo Cuenca Grau. Privacy in ontology-based information systems: A pending matter. *Semantic Web*, 1(1-2):137–141, 2010.

Bernardo Cuenca Grau and Boris Motik. Reasoning over Ontologies with Hidden Content: The Import-by-Query Approach. *Journal of Artificial Intelligence Research*, 45:197–255, 2012.

Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Modular reuse of ontologies: theory and practice. *Journal of Artificial Intelligence Research*, 31(1):273–318, 2008.

Volker Haarslev and Ralf Möller. Racer: A core inference engine for the semantic web. In *Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools*, volume 87 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2003.

Peter Haase, Vasant Honavar, Oliver Kutz, York Sure, and Andrei Tamilin. *Proceedings of the 1st International Workshop on Modular Ontologies, (WoMO 2006)*. CEUR-WS.org, 2006.

Joseph Y Halpern and Yoram Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial intelligence*, 54(3):319–379, 1992.

Leon Henkin. An extension of the Craig-Lyndon interpolation theorem. *Journal of Symbolic Logic*, pages 201–216, 1963.

Andreas Herzig and Jérôme Mengin. Uniform interpolation by resolution in modal logic. In *Logics in Artificial Intelligence, 11th European Conference, JELIA 2008*, volume 5293 of *Lecture Notes in Computer Science*, pages 219–231. Springer, 2008.

Matthew Horridge and Sean Bechhofer. The OWL API: A Java API for OWL ontologies. *Semantic Web*, 2(1):11–21, 2011.

Matthew Horridge, Bijan Parsia, and Ulrike Sattler. The state of bio-medical ontologies. *Bio-Ontologies 2011*, 2011.

Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical reasoning for very expressive description logics. *Logic Journal of the IGPL*, 8(3):239—263, 2000.

Ullrich Hustadt and Renate A. Schmidt. Issues of decidability for description logics in the framework of resolution. In Ricardo Caferra and Gernot Salzer, editors, *Automated Deduction in Classical and Non-Classical Logics*, volume 1761 of *Lecture Notes in Computer Science*, pages 191–205. Springer, 2000.

Ullrich Hustadt and Renate A Schmidt. Using resolution for testing modal satisfiability and building models. *Journal of Automated Reasoning*, 28(2):205–232, 2002.

Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Reducing $\mathcal{SHIQ}^-$ description logic to disjunctive datalog programs. In Didier Dubois, Christopher A. Welty, and Mary-Anne Williams, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004)*, pages 152—162. AAAI Press, 2004a.

Ullrich Hustadt, Renate A. Schmidt, and Lilia Georgieva. A survey of decidable first-order fragments and description logics. *Journal of Relational Methods in Computer Science*, 1(251-276):3, 2004b.

Yevgeny Kazakov. Consequence-Driven Reasoning for Horn $\mathcal{SHIQ}$ Ontologies. In Craig Boutilier, editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-09)*, pages 2040–2045. AAAI Press, 2009.

Yevgeny Kazakov and Boris Motik. A resolution-based decision procedure for $\mathcal{SHOIQ}$. *Journal of Automated Reasoning*, 40(2-3):89–116, 2008.

Yevgeny Kazakov, Markus Krötzsch, and František Simančík. ELK reasoner: Architecture and evaluation. In *Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012)*, volume 858 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

Yevgeny Kazakov, Markus Krötzsch, and František Simančík. The incredible ELK. *Journal of Automated Reasoning*, 53(1):1–61, 2014.

Jürg Kohlas, Rolf Haenni, and Serafín Moral. Propositional information systems. *Journal of Logic and Computation*, 9(5):651–681, 1999.

Boris Konev, Carsten Lutz, Dirk Walther, and Frank Wolter. Semantic modularity and module extraction in description logics. In *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008)*, volume 178 of *Frontiers in Artificial Intelligence and Applications*, pages 55–59. IOS Press, 2008a.

Boris Konev, Dirk Walther, and Frank Wolter. The logical difference problem for description logic terminologies. In *Automated Reasoning*, volume 5195 of *Lecture Notes of Computer Science*, pages 259–274. Springer, 2008b.

Boris Konev, Carsten Lutz, Dirk Walther, and Frank Wolter. Formal properties of modularisation. *Modular Ontologies*, 5445:25–66, 2009a.

Boris Konev, Dirk Walther, and Frank Wolter. Forgetting and uniform interpolation in large-scale description logic terminologies. In *Proceedings of the Internation Conference on Artificial Intelligence (IJCAI-09)*, pages 830–835. AAAI Press, 2009b.

Boris Konev, Michel Ludwig, Dirk Walther, and Frank Wolter. The logical difference for the lightweight description logic $\mathcal{EL}$. *Journal of Artificial Intelligence Research*, 44(1):633–708, 2012.

Boris Konev, Carsten Lutz, Dirk Walther, and Frank Wolter. Model-theoretic inseparability and modularity of description logic ontologies. *Artificial Intelligence*, 203: 66–103, 2013.

Roman Kontchakov, Frank Wolter, and Michael Zakharyaschev. Logic-based ontology comparison and module extraction, with an application to DL-Lite. *Artificial Intelligence*, 174(15):1093–1141, 2010.

Patrick Koopmann and Renate A. Schmidt. Uniform interpolation of $\mathcal{ALC}$-ontologies using fixpoints. In *Frontiers of Combining Systems*, volume 8152 of *Lecture Notes in Computer Science*, pages 87–102. Springer, 2013a.

Patrick Koopmann and Renate A. Schmidt. Implementation and evaluation of forgetting in $\mathcal{ALC}$-ontologies. In *Proceedings of the 7th International Workshop on Modular Ontologies (WoMO'13)*, volume 1081 of *CEUR Workshop Proceedings*, pages 37–48. CEUR-WS.org, 2013b.

Patrick Koopmann and Renate A. Schmidt. Forgetting concept and role symbols in $\mathcal{ALCH}$-ontologies. In *Logic for Programming, Artificial Intelligence and Reasoning*, volume 8312 of *Lecture Notes in Computer Science*, pages 552–567. Springer, 2013c.

Patrick Koopmann and Renate A. Schmidt. Count and forget: Uniform interpolation of $\mathcal{SHQ}$-ontologies. In *Automated Reasoning*, volume 8562 of *Lecture Notes in Computer Science*, pages 434–448. Springer, 2014a.

Patrick Koopmann and Renate A. Schmidt. Computing uniform interpolants of $\mathcal{ALCH}$-ontologies with background knowledge. In *Joint Automated Reasoning Workshop and Deduktionstreffen*, 2014b.

Patrick Koopmann and Renate A. Schmidt. Forgetting and uniform interpolation for $\mathcal{ALC}$-ontologies with ABoxes. In *Proceedings of the 27th International Workshop of Description Logics (DL 2014)*, volume 1193 of *CEUR Workshop Proceedings*, pages 245–257. CEUR-WS.org, 2014c.

Patrick Koopmann and Renate A. Schmidt. Uniform interpolation and forgetting for $\mathcal{ALC}$ ontologies with ABoxes. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-2015)*, pages 175–181. AAAI-Press, 2015a.

Patrick Koopmann and Renate A. Schmidt. Saturation-based forgetting in the description logic $\mathcal{SIF}$. In *Proceedings of the 28th International Workshop of Description Logics (DL 2015)*, volume 1350 of *CEUR Workshop Proceedings*, pages 439–451. CEUR-WS.org, 2015b.

Patrick Koopmann and Renate A. Schmidt. LETHE: Saturation-based reasoning for non-standard reasoning tasks. In *Informal Proceedings of the 4th International Workshop on OWL Reasoner Evaluation (ORE-2015)*, volume 1387 of *CEUR Workshop Proceedings*, pages 23–30. CEUR-WS.org, 2015c.

Marcus Kracht. Modal consequence relations. In Frank Wolter Patrick Blackburn, Johan van Benthem, editor, *Handbook of Modal Logic*, volume 3 of *Studies in Logic and Practical Reasoning*, chapter 8. Elsevier, 2007.

Jérôme Lang and Pierre Marquis. Resolving inconsistencies by variable forgetting. In *Principles of Knowledge Representation And Reasoning - International Conference*, pages 239–250. Morgan Kaufmann, 2002.

Jérôme Lang, Paolo Liberatore, and Pierre Marquis. Propositional independence. *Journal of Artificial Intelligence Research*, 18:391–443, 2003.

Fangzhen Lin. On strongest necessary and weakest sufficient conditions. *Artificial Intelligence*, 128(1):143–159, 2001.

Fangzhen Lin and Ray Reiter. Forget it! In *Working Notes of AAAI Fall Symposium on Relevance*, pages 154–159. AAAI Press, 1994.

Hongkai Liu, Carsten Lutz, Maja Miličić, and Frank Wolter. Foundations of instance level updates in expressive description logics. *Artificial Intelligence*, 175(18):2170–2197, 2011. ISSN 0004-3702.

Michael Ludwig and Boris Konev. Practical uniform interpolation and forgetting for $\mathcal{ALC}$ TBoxes with applications to logical difference. In *Proceedings, Fourteenth International Conference on Principles of Knowledge Representation and Reasoning (KR-14)*. AAAI Press, 2014.

Michel Ludwig and Boris Konev. Towards practical uniform interpolation and forgetting for $\mathcal{ALC}$ TBoxes. In *Informal Proceedings of the 26th International Workshop on Description Logics*, volume 1014 of *CEUR Workshop Proceedings*, pages 377–389. CEUR-WS.org, 2013.

Carsten Lutz and Frank Wolter. Conservative extensions in the lightweight description logic $\mathcal{EL}$. In *Automated Deduction–CADE-21*, pages 84–99. Springer, 2007.

Carsten Lutz and Frank Wolter. Deciding inseparability and conservative extensions in the description logic $\mathcal{EL}$. *Journal of Symbolic Computation*, 45(2):194–228, 2010.

Carsten Lutz and Frank Wolter. Foundations for uniform interpolation and forgetting in expressive description logics. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-11)*, pages 989–995. AAAI Press, 2011.

Carsten Lutz, Dirk Walther, and Frank Wolter. Conservative extensions in expressive description logics. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence*, volume 7, pages 453–458. AAAI Press, 2007.

Carsten Lutz, Robert Piro, and Frank Wolter. Enriching $\mathcal{EL}$-concepts with greatest fixpoints. In *Proceedings of the 2010 conference on ECAI 2010: 19th European Conference on Artificial Intelligence*, pages 41–46. IOS Press, 2010.

Carsten Lutz, Inanç Seylan, and Frank Wolter. An automata-theoretic approach to uniform interpolation and approximation in the description logic $\mathcal{EL}$. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference (KR-12)*, pages 286–296. AAAI Press, 2012.

Nicolas Matentzoglu, Samantha Bail, and Bijan Parsia. A snapshot of the OWL web. In *The Semantic Web - ISWC 2013*, volume 8218 of *Lecture Notes of Computer Science*, pages 331–346. Springer, 2013.

Grigori Mints. Resolution calculi for modal logics. *American Mathematical Society Translations*, 143:1–14, 1989.

Grigori Mints. Gentzen-type systems and resolution rules part I: propositional logic. In *COLOG-88: International Conference on Computer Logic, Tallinn, USSR*, volume 417 of *Lecture Notes in Computer Science*, pages 198–231. Springer Berlin Heidelberg, 1990.

Boris Motik, Rob Shearer, and Ian Horrocks. Hypertableau reasoning for description logics. *Journal of Artificial Intelligence Research*, 36(1):165–228, 2009.

Claudia Nalon and Clare Dixon. Clausal resolution for normal modal logics. *Journal of Algorithms*, 62(34):117 – 134, 2007.

Cláudia Nalon, João Marcos, and Clare Dixon. Clausal resolution for modal logics of confluence. In *Automated Reasoning*, volume 8562 of *Lecture Notes of Computer Science*, pages 322–336. Springer, 2014.

Nadeschda Nikitina. Forgetting in general $\mathcal{EL}$ terminologies. In *Proceedings of the 2011 International Workhop on Description Logics (DL2011)*, volume 745 of *CEUR Workshop Proceedings*, pages 345–355. CEUR-WS.org, 2011.

Nadeschda Nikitina and Birte Glimm. Hitting the Sweetspot: Economic Rewriting of Knowledge Bases. In *The Semantic Web: ISWC-12*, volume 7649 of *Lecture Notes in Computer Science*, pages 394–409. Springer, 2012.

Nadeschda Nikitina and Sebastian Rudolph. ExpExpExplosion: Uniform interpolation in general $\mathcal{EL}$ terminologies. In *ECAI 2012: Proceedings of the 20th European Conference on Artificial Intelligence*, Frontiers in Artificial Intelligence and Applications. IOS Press, 2012.

Nadeschda Nikitina and Sebastian Rudolph. (Non-)Succinctness of uniform interpolants of general terminologies in the description logic $\mathcal{EL}$. *Artificial Intelligence*, 215:120–140, 2014.

Nadeschda Nikitina and Sven Schewe. More is sometimes less: Succinctness in $\mathcal{EL}$. In *Informal Proceedings of the 26th International Workshop on Description Logics*, volume 1014 of *CEUR Workshop Proceedings*, pages 403–414. CEUR-WS.org, 2013a.

Nadeschda Nikitina and Sven Schewe. Simplifying description logic ontologies. In *The Semantic Web–ISWC 2013*, pages 411–426. Springer, 2013b.

Andreas Nonnengart and Andrzej Szałas. A fixpoint approach to second order quantifier elimination with applications to correspondence theory. Technical report, Max-Planck-Institut für Informatik, 1995.

Andreas Nonnengart and Christoph Weidenbach. Computing small clause normal forms. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, volume 1 of *Handbook of Automated Reasoning*, chapter 6, pages 335–367. North-Holland, Amsterdam, 2001.

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*, 37:170–173, 2009.

Martin Odersky, Philippe Altherr, Vincent Cremet, Burak Emir, Sebastian Maneth, Stéphane Micheloud, Nikolay Mihaylov, Michel Schinz, Erik Stenman, and Matthias Zenger. An overview of the Scala programming language. Technical report, École Polytechnique Fédérale de Lausanne, 2004.

Hans Jürgen Ohlbach. A resolution calculus for modal logics. In *9th International Conference on Automated Deduction*, pages 500–516. Springer, 1988.

Hans Jürgen Ohlbach. SCAN—elimination of predicate quantifiers. In *Automated Deduction: CADE-13*, volume 1104 of *Lecture Notes of Computer Science*, pages 161–165. Springer, 1996.

Magdalena Ortiz, Sebastian Rudolph, and Mantas Simkus. Worst-case optimal reasoning for the Horn-DL fragments of OWL 1 and 2. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference (KR-10)*, 2010.

Jeff Z. Pan and Edward Thomas. Approximating OWL-DL ontologies. In *Proceedings of the Twentieth AAAI Conference on Artificial Intelligence (AAAI-07)*, pages 1434–1439. AAAI Press, 2007.

Gabrielle Paul. Approaches to abductive reasoning: an overview. *Artificial intelligence review*, 7(2):109–152, 1993.

Charles S. Peirce. Deduction, induction, and hypothesis. *Popular science monthly*, 13: 470–482, 1878.

Daniel Pokrywczynski and Dirk Walther. Deciding the logical difference problem for $\mathcal{EL}$ with role inclusions. volume 348 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

Guilin Qi and Jianfeng Du. Model-based revision operators for terminologies in description logics. In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI-09)*, pages 891–897. IJCAI Organization/AAAI Press, 2009.

Guilin Qi, Yimin Wang, Peter Haase, and Pascal Hitzler. A forgetting-based approach for reasoning with distributed ontologies. In Ulrike Sattler and Andrei Tamilin, editors, *Proceedings of the Workshop on Ontologies: Reasoning and Modularity (WORM'08)*, volume 348. CEUR, 2008.

Yuan Ren, Jeff Z. Pan, and Yuting Zhao. Soundness preserving approximation for TBox reasoning. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI-10)*, pages 351–356. AAAI Press, 2010.

A. Riazanov and A. Voronkov. The design and implementation of VAMPIRE. *AI communications*, 15(2):91–110, 2002.

Ana Armas Romero, Bernardo Cuenca Grau, and Ian Horrocks. MORe: Modular combination of OWL reasoners for ontology classification. In *The Semantic Web–ISWC 2012*, pages 1–16. Springer, 2012.

Ana Armas Romero, Mark Kaminski, Bernardo Cuenca Grau, and Ian Horrocks. Ontology module extraction via datalog reasoning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 1410–1416, 2015.

Ulrike Sattler, Thomas Schneider, and Michael Zakharyaschev. Which kind of module should I extract? In *Proceedings of the 22nd International Workshop on Description Logics (DL 2009)*, volume 477 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

Renate A Schmidt. Resolution is a decision procedure for many propositional modal logics. In *Advances in Modal Logic*, volume 87 of *Lecture Notes*, pages 189–208. Citeseer, 1996.

Renate A Schmidt. Decidability by resolution for propositional modal logics. *Journal of Automated Reasoning*, 22(4):379–396, 1999.

Renate A. Schmidt. A new methodology for developing deduction methods. *Annals of Mathematics and Artificial Intelligence*, 55(1-2):155–187, 2009.

Renate A. Schmidt. The Ackermann approach for modal logic, correspondence theory and second-order reduction. *Journal of Applied Logic*, 10(1):52–74, 2012.

Renate A. Schmidt and Ullrich Hustadt. A principle for incorporating axioms into the first-order translation of modal formulae. In *Automated Deduction–CADE-19*, pages 412–426. Springer, 2003a.

Renate A. Schmidt and Ullrich Hustadt. Mechanised reasoning and model generation for extended modal logics. In *Theory and Applications of Relational Structures as Knowledge Instruments*, volume 2929 of *Lecture Notes of Computer Science*, pages 38–67. Springer, 2003b.

Stephan Schulz. E—a brainiac theorem prover. *AI Communications*, 15(2):111–126, 2002.

Rob Shearer, Boris Motik, and Ian Horrocks. HermiT: A highly-efficient OWL reasoner. In *OWL: Experiences and Directions (OWLED'08)*, pages 26–27. CEUR-WS.org, 2008.

František Simančík, Yevgeny Kazakov, and Ian Horrocks. Consequence-based reasoning beyond Horn ontologies. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI-11)*, volume 22, pages 1093–1098. AAAI Press, 2011a.

František Simančík, Boris Motik, and Markus Krötzsch. Fixed parameter tractable reasoning in DLs via decomposition. In *Proceedings of the 24th International Workshop on Description Logics (DL 2011)*, volume 745 of *CEUR Workshop Proceedings*, pages 400–410. CEUR-WS.org, 2011b.

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.

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.

Michael Q. Stearns, Colin Price, Kent A. Spackman, and Amy Y. Wang. SNOMED

clinical terms: overview of the development process and project status. In *Proceedings of the AMIA Symposium*, pages 662–666. American Medical Informatics Association, 2001.

Andreas Steigmiller, Birte Glimm, and Thorsten Liebig. Coupling tableau algorithms for expressive description logics with completion-based saturation procedures. In *Automated Reasoning*, volume 8562 of *Lecture Notes of Computer Science*, pages 449–463. Springer, 2014a.

Andreas Steigmiller, Thorsten Liebig, and Birte Glimm. Konclude: system description. *Web Semantics: Science, Services and Agents on the World Wide Web*, 27:78–85, 2014b.

Heiner Stuckenschmidt, Christine Parent, and Stefano Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *Lecture Notes in Computer Science*. Springer, 2009.

Andrzej Szalas. On the correspondence between modal and classical logic: An automated approach. *Journal of Logic and Computation*, 3(6):605–620, 1993.

Balder ten Cate, Willem Conradie, Maarten Marx, and Yde Venema. Definitorially complete description logics. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Tenth International Conference (KR-06)*, pages 79–89. AAAI Press, 2006.

Stephan Tobies. *Complexity results and practical algorithms for logics in knowledge representation*. PhD thesis, LuFG Theoretical Computer Science, RWTH-Aachen, Germany, 2001.

Dmitry Tsarkov and Ignazio Palmisano. Chainsaw: a metareasoner for large ontologies. In *Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE 2012)*, volume 858 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

Albert Visser. Uniform interpolation and layered bisimulation. In *Gödel'96: Logical foundations of mathematics, computer science and physics—Kurt Gödel's legacy*, pages 139–164. Association for Symbolic Logic, 1996.

Kewen Wang, Zhe Wang, Rodney Topor, Jeff Pan, and Grigoris Antoniou. Concept and role forgetting in $\mathcal{ALC}$ ontologies. In *The Semantic Web - ISWC 2009*, volume 5823 of *Lecture Notes of Computer Science*, pages 666–681. Springer, 2009.

Kewen Wang, Zhe Wang, Rodney W. Topor, Jeff Z. Pan, and Grigoris Antoniou. Eliminating concepts and roles from ontologies in expressive descriptive logics. *Computational Intelligence*, 30(2):205–232, 2014.

Zhe Wang, Kewen Wang, Rodney Topor, and Jeff Z. Pan. Forgetting concepts in DL-Lite. In Sean Bechhofer, Manfred Hauswirth, Jörg Hoffmann, and Manolis Koubarakis, editors, *The Semantic Web: Research and Applications*, volume 5021 of *Lecture Notes in Computer Science*, pages 245–257. Springer, 2008.

Zhe Wang, Kewen Wang, Rodney Topor, and Xiaowang Zhang. Tableau-based forgetting in $\mathcal{ALC}$ ontologies. In *ECAI 2010 Proceedings*, pages 47–52. IOS Press, 2010a.

Zhe Wang, Kewen Wang, Rodney W. Topor, and Jeff Z. Pan. Forgetting for knowledge bases in DL-Lite. *Annals of Mathematics and Artificial Intelligence*, 58(1–2):117–151, 2010b.

Christoph Weidenbach, Uwe Brahm, Thomas Hillenbrand, Enno Keen, Christian Theobald, and Dalibor Topić. SPASS version 2.0. In *Automated Deduction: CADE-18*, pages 275–279. Springer, 2002.

Yan Zhang and Yi Zhou. Forgetting revisited. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference (KR-10)*, pages 602–604. AAAI Press, 2010.

Yi Zhou and Yan Zhang. Bounded forgetting. In Wolfram Burgard and Dan Roth, editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-11)*, pages 280–285. AAAI Press, 2011.