Bounded Model Reasoning For The Semantic Web

MASTER THESIS

Submitted on: 22. April 2015

by: Lukas Schweizer
Student-ID 3899842

Supervisor: Prof. Dr. rer. nat. Sebastian Rudolph
Faculty of Computer Science, Institute of Artificial Intelligence
Professorship Computational Logic

TECHNISCHE UNIVERSITÄT DRESDEN

Declaration

I hereby declare, that except where specific reference is made to the work of others, the contents of this thesis are original and have not been submitted in whole, or in part to obtain a degree or any other qualification neither in this nor in any other university. This thesis is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text. No other resources apart from the references and auxiliary means indicated in the bibliography were used in the development of the presented work.

(Place, Date)	Lukas Schweizer George-Bähr-Straße, 14, 01069 Dresden

Abstract

With increasing success of *Semantic Web Technologies*, in particular the *Web Ontology Language* (OWL), these technologies naturally are now and then used in a way they originally were not meant to be used and applied. We are motivated by the observation, that OWL is used to formalize an application domain in a self-contained manner, and thereby might embody some sort of *Constraint Satisfaction Problem*. We argue, that in these cases the standard semantics does not really comply with the actual envisioned models – namely solutions of the encoded problem.

We first modify the standard semantics by means of restricting the domain to be finite and of a priori fixed size, induced by those individuals occurring in the given knowledge base of interest – the *Bounded Model Semantics*. We are focus on the description logic $\mathcal{SROIQ}$, as logical basis of OWL, in which it is possible to impose an additional axiom enforcing any existing reasoner to perform bounded model reasoning. Whereas reasoning in $\mathcal{SROIQ}$ is N2ExpTime-complete, we show that reasoning with respect to bounded models is NP-complete. Though, existing reasoners potentially struggle on bounded model reasoning task, as they are not designed for such combinatoric tasks.

In consequence we propose a quiet different approach, introducing a translation of $\mathcal{SROIQ}$ knowledge bases into normal logic programs under the *answer set semantics*. We show that the set of bounded models coincides with the set of answer sets of the obtained program, allowing us to use existing answer set solvers and employ them to our needs of *Bounded Model Reasoning*. We benefit from this approach in way that we can also request all models, which we nominated as additional non-standard reasoning task called *model enumeration*. We have implemented the approach, for which an initial evaluation supports our theoretical findings, and gives confidence in claiming, that our approach is a fertile combination of two knowledge representation formalisms.

Contents

1 Introduction

1.1 Motivation and Problem Description

Semantic Web Technologies reached mainstream audience due to well standardized specifications and well developed tool line-ups, easily accessible and usable without having extensive training in foundations of computer science. At the sharp end, this accounts mainly to the *Web Ontology Language* OWL, the most prominent formal language available for ontological modeling, developed and standardized by the W3C [W3C09]. It is designed to enable *domain experts* expressing their *domain knowledge* in a precise manner and sharable via the Web with the ulterior motive of machine accessible knowledge content [W3C09]. We want to motivate our work with an example.

Example 1.1 (Graduate Study Problem)**.** *The study regulations of some graduate master program somewhere in the academic world, specifies all necessary steps, a student must acquire in order to graduate. Therefore, beside obligatory courses, students are able to choose freely the contents of modules of which they must complete three during their studies. Available modules are,*

- *Knowledge Representation and Reasoning (KRR),*
- *Theoretical Computer Science (ThCS),*
- *Principles of Inference (PI), and*
- *Advanced Logic (AL).*

Now there is a global list of courses offered at the educational institute, whereas each course is built up on either 2, 4, 6 or 8 hours of lecturing per week. Students who took courses summing up to 8 hours of lecturing per week are able to complete exactly one of those modules, which comprises these courses. It is easier to illustrate the problem in form of the following cross-table, as Table 1.1 does, listing courses and their weekly amount of lecturing as well as indicating (via X) in what module the course can be used. The task is to take courses, such that three modules can be completed, where a course can only be used once.

Example 1.1 embodies the knowledge domain of an institutional master program as it is offered typically at universities. Suppose now, that the person in charge of specifying this regulation is entirely convinced of the idea, that formulating his knowledge using OWL as logic-based representation formalism and distributing final versions of the regulations publicly on the Internet is a win-win situation for everybody involved. Making available precise definitions of compositions of courses into modules, statements

	Hours	KRR	ThCS	PI	AL
Description Logics	6	X	X	X	
Advanced Logic	8				X
Complexity Theory	4		X		
Abstract Argumentation	2	X		X	
Formal Concept Analysis	4	X		X	
Semantic Web Technologies	6	X		X	
Automata and Logic	6	X	X	X	
Term Rewriting Systems	6		X		
KRR Seminar	2	X		X	
ThCS Seminar	2	X	X	X	

Table 1.1: Graduate Study Problem – Course Listing.

(axioms) denoting possible course combinations, or others excluding impossible course enrollments, additional information on lectures, and so forth. In this way, the elaborated study regulations can be tested on inconsistencies; i.e. the study regulations do not contain contradictory statements, leading to loopholes potentially identified by resourceful students, for example. Also other applications of his institute benefit, for example, the regulation directly can be accessed by students on the institute's Web page. In short, and to a certain extend, this is what is envisioned with the idea of the *Semantic Web*.

Motivation Suppose now that students preferably would like to have a supporting system along their studies guiding them in solving the imposed problem of course and module combinations; i.e. a system providing concrete suggestions on how to complete modules by means of selecting appropriate courses. Suddenly, we can perceive the study regulations as some form of *constraint-satisfaction* problem, where students are not only interested whether their study regulations are consistent (they should anyway), but rather in solutions of the imposed problem. We want to investigate into these circumstances, where given knowledge bases embody some certain constraint-satisfaction type. As in our example, these knowledge bases then typically specify the underlying problem in full detail, which means that all information in order to solve the problem is given and present. In some way, we could say that such knowledge bases are expected to be self-contained and represent an insular knowledge domain.

Problem Description In these scenarios, the standard semantics do not really comply with, which elicits the need for more intuitive semantics. Solutions of the problem represented in some knowledge base could therefore be considered as more intuitive models, rather than arbitrary ones. In consequence, we first impose the problem of defining adequate semantics under the condition, of not totally faulting the standard semantics, but rather careful restrictions; i.e. remain in monotonic waters. Further, we identify

the need of efficient tool support. In particular, we require additional functionality to obtain all potential solutions – a task we will throughout this work refer to as *model enumeration*.

1.2 Outline

In short, readers who feel familiar with *description logics* and $\mathcal{SROIQ}$ in particular, as well as with the notions in the context of *answer set programming*, can safely skip Chapter 2. We start with our own new notions in Chapter 3, which together with the subsequent ones represent the core work of this thesis.

Chapter 2 The required theoretical background is introduced. Beginning with defining syntax and semantics of the *description logic* $\mathcal{SROIQ}$, illustrated with examples, and followed by important properties the logic enjoys. Subsequently, in Section 2.2 the theory of logic programs evaluated under the *answer set semantics* are presented, further known as *answer set programming* representing a modern approach towards true declarative programming.

Chapter 3 In this chapter, we introduce the notion of *bounded models* and reasoning with respect to such models. We identify the new reasoning tasks *model extraction* and *model enumeration* and show that testing satisfiability with respect to bounded models is NP-complete. Moreover, it is also demonstrated how existing tableaux based reasoners which perform exceptionally well on standard reasoning, potentially tend to struggle when enforced to reason with respect to bounded models.

Chapter 4 An encoding of any $\mathcal{SROIQ}$ knowledge base $\mathcal{K}$, into an answer set program $\Pi(\mathcal{K})$ is successively developed in this chapter. Substantially, we show that the set of answer sets of $\Pi(\mathcal{K})$ coincides with the set of bounded models of $\mathcal{K}$. This allows us to perform not only standard reasoning tasks on $\mathcal{K}$, but also the proposed tasks of *model extraction* and *enumeration* quite elegantly.

Chapter 5 Based on the elaborated results, we present Wolpertinger, a *bounded model reasoner* and lightweight implementation of our proposed approach. After describing its system architecture, including all main components and their interdependencies, we conduct an initial evaluation which underpin our theoretical findings and in general leaves a promising impression.

Chapter 6 We conclude with Chapter 6, first providing a retrospective summary, leading to some general conclusions and assessment of our approach. We round up with an overview on related work, in Section 6.1, and lastly an outlook in Section 6.2 on potential future work.

2 Formal Preliminaries

In this chapter the required theoretical background is introduced. Beginning with the description logic $\mathcal{SROIQ}$, which is the logical basis of OWL 2. Subsequently, in Section 2.2, the theory of logic programs evaluated under the answer set semantics, consequently called answer set program, a modern approach towards true declarative programming.

2.1 The Description Logic $\mathcal{SROIQ}$

Description Logics (DLs) are a family of knowledge representation formalisms, designed to represent the terminological knowledge of an application domain in a formally well-understood way [BCM+07]. While DLs are already embodied in several knowledge-based systems, $\mathcal{SROIQ}(\mathbf{D})$ certainly is the most famous representative, since it serves as the logical basis of the OWL 2.[1] However, in this thesis we will only work with $\mathcal{SROIQ}$, and therefore not consider concrete domains.

2.1.1 Syntax

The following definitions comply with the ones given in [HKS06] and [Rud11]. Comprehensive foundations in description logics are available in [BCM+07].

Definition 2.1 ($\mathcal{SROIQ}$ vocabulary)**.** *Let N_C^{dl}, N_R^{dl} and N_I^{dl} be three disjoint sets, N_C^{dl} be the set of concept names, N_R^{dl} be the set of role names including the universal role u, and N_I^{dl} is the set of individual names. The set of roles is $\mathbf{R} := N_R^{dl} \cup \{r^- |\ r \in N_R^{dl}\}$, where a role r^- is called the inverse role of r. In the sequel, r and s denote roles, possibly indexed.*

The function Inv is defined on roles, sucht that $\mathsf{Inv}(r) = r^- \in \mathbf{R}$ if $r \in \mathbf{R}$ is a role name, and $\mathsf{Inv}(r) = s \in \mathbf{R}$, if $r = s^-$.

RBox A role box $\mathcal{R}$ consists of two components, a finite set $\mathcal{R}_h$ of *(generalized) role inclusion axioms*, called *role hierarchy*, and a finite set of *role assertions* $\mathcal{R}_a$.

Definition 2.2 ((Regular) Role Inclusion Axioms)**.** *A generalized role inclusion axiom (RIA for short) is an expression of the form $s_1 \circ \cdots \circ s_n \sqsubseteq r$, with $s_i \neq u$ for $1 \leq i \leq n$, and $r \neq u$ is a role name. A role hierarchy $\mathcal{R}_h$ is a finite set of RIAs. $\mathcal{R}_h$ is called regular, if there is a strict partial order $\prec$ on $\mathbf{R}$, such that*

[1] More precisely, it is the logical basis of OWL DL, the fragment of OWL 2 which purely relies on $\mathcal{SROIQ}(\mathbf{D})$.

1. $s \prec r$ *iff* $\mathsf{Inv}(s) \prec r$, *and*
2. *every RIA is of the form*
   - $r \circ r \sqsubseteq r$, *or*
   - $r^- \sqsubseteq r$, *or*
   - $s_1 \circ \cdots \circ s_n \sqsubseteq r$, *or*
   - $r \circ s_1 \circ \cdots \circ s_n \sqsubseteq r$, *or*
   - $s_1 \circ \cdots \circ s_n \circ r \sqsubseteq r$

   where $r \in N_R^{dl}$ *is a (non-inverse) role name, and* $s_i \prec r$ *for* $1 \leq i \leq n$.

The set of simple-roles $\mathbf{R}^s$ *for some role hierarchy is defined inductively:*

- *if a role* r *occurs only on the right hand-side of RIAs of the form* $r \sqsubseteq s$ *such that* s *is simple, then* r *is also simple.*
- *the inverse of some simple role is also simple.*

The set of non-simple roles therefore is denoted by $\mathbf{R}^n := \mathbf{R} \setminus \mathbf{R}^s$.

Regularity ensures decidability when dealing with $\mathcal{SROIQ}$ RBoxes, while arbitrary role chains could lead to undecidable results. The notion of non-simple roles enables to distinguish between those roles which can be composed by role chains of length greater than one, and those which can not.

Definition 2.3 (Role Assertions)**.** *For roles* r,s, $s' \neq u$, *the assertions* $\mathsf{Ref}(r)$, $\mathsf{Irr}(r)$, $\mathsf{Sym}(r)$, $\mathsf{Asy}(r)$, $\mathsf{Tra}(r)$, *and* $\mathsf{Dis}(s, s')$ *are called role assertions, where* s *and* s' *are simple.*

TBox Terminological knowledge, defining interdependencies between concepts, is captured in the TBox (*terminological box*). Having the vocabulary fixed, concepts can be described by means of *concept constructors*. The set of constructors considered in $\mathcal{SROIQ}$ are the *bottom concept* ($\bot$), *universal concept* ($\top$), *conjunction* ($\sqcap$), *disjunction* ($\sqcup$), *negation* ($\neg$), *value restriction* ($\forall r.C$), *existential restriction* ($\exists r.C$), *self restriction* ($\exists s.Self$), *at-least restriction* ($\geq n\ r.C$), *at-most restriction* ($\leq n\ r.C$), as well as the *nominal constructor* ($\{\cdot\}$).

Definition 2.4 (Concept description, TBox)**.** *Given a* $\mathcal{SROIQ}$ *RBox* $\mathcal{R}$, *the set of concept descriptions (or simply concepts) is the smallest set* $\mathbf{C}$, *such that*

1. $\top \in \mathbf{C}$, $\bot \in \mathbf{C}$ *as well as* $N_C^{dl} \subseteq \mathbf{C}$;
2. *for every* $\{a_1, \ldots, a_n\} \subseteq N_I^{dl}$, $\{a_1, \ldots, a_n\} \in \mathbf{C}$;
3. *if* C,$D \in \mathbf{C}$ *are concepts,* $r \in \mathbf{R}$ *is a role (possibly inverse),* $s \in \mathbf{R}$ *is a simple role (possibly inverse), and* n *is a non-negative integer, then* $C \sqcap D$, $C \sqcup D$, $\neg C$, $\forall r.C$, $\exists r.C$, $\exists s.Self$, $(\geq n\ s.C)$, *and* $(\leq n\ s.C)$ *are concepts as well.*

A general concept inclusion axiom (GCI) is of the form $C \sqsubseteq D$, *with* C, $D \in \mathbf{C}$. *A* $\mathcal{SROIQ}$ *TBox* $\mathcal{T}$ *is a finite set of GCIs.*

ABox Assertional knowledge, representing facts of the underlying knowledge domain, is captured in the ABox (*assertional box*).

Definition 2.5 (Individual Assertion, ABox)**.** *Let a,$b \in N_I^{dl}$ individual names, $C \in \mathbf{C}$ a concept description, $r \in \mathbf{R}$ a role (possibly simple). An individual assertion then can have any of the following forms:*

- *$C(a)$, called concept assertion;*
- *$r(a,b)$ or $\neg r(a,b)$, called (negated) role assertion;*
- *$a \approx b$, called equality statement, or*
- *$a \not\approx b$, called inequality statement;*

A $\mathcal{SROIQ}$ ABox $\mathcal{A}$ is a finite set of individual assertions.

Knowledge Base A $\mathcal{SROIQ}$ knowledge base $\mathcal{K}$ is the union of a regular RBox $\mathcal{R}$ and a TBox $\mathcal{T}$ as well as an ABox $\mathcal{A}$ for $\mathcal{R}$. The elements of $\mathcal{K}$ are referred to as *axioms*. Given a knowledge base $\mathcal{K}$ the sets $N_I^{dl}(\mathcal{K})$, $N_C^{dl}(\mathcal{K})$, and $N_R^{dl}(\mathcal{K})$ denote those individual names, concept names, and role names which occur in $\mathcal{K}$, respectively.

Example 2.6 (Graduate Study Problem)**.** *We can formulate now some aspects of the GSP illustrated in Example 1.1, by means of a $\mathcal{SROIQ}$ knowledge base $\mathcal{K}_1 = (\mathcal{R}_1, \mathcal{T}_1, \mathcal{A}_1)$, and start with TBox $\mathcal{T}_1$:*

$$
\begin{aligned}
\mathcal{T}_1 \;=\; \{ \quad \mathsf{Student} &\sqsubseteq \geq 3\ \mathsf{completed.Module}, && (2.1)\\
\mathsf{Module} &\equiv \{\mathsf{krr}\} \sqcup \{\mathsf{thcs}\} \sqcup \{\mathsf{advlogic}\} \sqcup \{\mathsf{pi}\}, && (2.2)\\
\mathsf{Module} &\sqsubseteq \exists \mathsf{offers.Course}, && (2.3)\\
\mathsf{Course} &\equiv \mathsf{2HCourse} \sqcup \mathsf{4HCourse} \sqcup \mathsf{6HCourse} \sqcup \mathsf{8HCourse}, && (2.4)\\
\exists \mathsf{completed.}\{\mathsf{krr}\} &\sqsubseteq \geq 2\ \mathsf{passed.}(\mathsf{4HCourse} \sqcap (\exists \mathsf{offers}^{-}.\{\mathsf{krr}\})) \sqcup \exists \mathsf{passed.}(\mathsf{8HCourse} \sqcap (\exists \mathsf{offers}^{-}, \{\mathsf{krr}\})) && (2.5)\\
\}
\end{aligned}
$$

Where, for example, Axiom (2.1) states that any student necessarily has completed at least three modules, and Axiom (2.2) defines any module to be one of the ones in $\{krr, thcs, advlogic, pi\}$. To complete the module krr, it is required by Axiom (2.5), to pass at least two courses of four hour lecturing, offered by krr, or simply pass an eight hour course offered by krr. An ABox $\mathcal{A}_1$ might look as follows:

$$
\begin{aligned}
\mathcal{A}_1 \;=\; \{ \quad & \mathsf{Student}(lukas),\ \mathsf{4HCourse}(complexity),\ \mathsf{4HCourse}(ifca),\\
& \mathsf{8HCourse}(advlogic),\ \mathsf{offers}(krr, complexity),\ \mathsf{offers}(krr, ifca) \quad \}
\end{aligned}
$$

Promoting named individual lukas to be a student, and, for example, denoting ifca to be a four hour course offered in the module krr.

2.1.2 Semantics

Definition 2.7 (Interpretation)**.** *An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty domain $\Delta^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$, mapping*

- *every individual $a \in N_I^{dl}$ to a corresponding element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$;*
- *every concept name $A \in N_C^{dl}$ to a corresponding set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$;*
- *every role name $r \in N_R^{dl}$ to a corresponding set $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$;*

Definition 2.8 (Semantics, Satisfiability)**.** *Given an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, let $u^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, then for concepts C, D, roles r, s, and non-negative integers n, the function $\cdot^{\mathcal{I}}$ is extended to concepts inductively as follows:*

$$
\begin{aligned}
\top^{\mathcal{I}} = \Delta^{\mathcal{I}} \quad & \bot^{\mathcal{I}} = \emptyset \quad (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \quad & (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(r^-)^{\mathcal{I}} &= \{(y,x) \mid (x,y) \in r^{\mathcal{I}}\} \\
(\exists r.C)^{\mathcal{I}} &= \{x \mid \exists y.(x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \\
(\exists r.Self)^{\mathcal{I}} &= \{x \mid (x,x) \in r^{\mathcal{I}}\} \\
(\forall r.C)^{\mathcal{I}} &= \{x \mid \forall y.(x,y) \in r^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\} \\
(\leq n\ r.C)^{\mathcal{I}} &= \{x \mid \#\{y \mid (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \leq n\} \\
(\geq n\ r.C)^{\mathcal{I}} &= \{x \mid \#\{y \mid (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \geq n\} \\
\{a_1, \ldots, a_n\}^{\mathcal{I}} &= \{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\}
\end{aligned}
$$

$\mathcal{I}$ satisfies an axiom $\alpha \in \mathcal{K}$, if $\mathcal{I} \models \alpha$; $\mathcal{I}$ is then called a model of α. For an RBox $\mathcal{R}$ and its axioms this yields:

- *$\mathcal{I} \models s \sqsubseteq r$, if $s^{\mathcal{I}} \subseteq r^{\mathcal{I}}$;*
- *$\mathcal{I} \models s_1 \circ \cdots \circ s_n \sqsubseteq r$, if $s_1^{\mathcal{I}} \circ \cdots \circ s_n^{\mathcal{I}} \subseteq r^{\mathcal{I}}$;*
- *$\mathcal{I} \models \mathsf{Ref}(r)$, if $\{(x,x) \mid x \in \Delta^{\mathcal{I}}\} \subseteq r^{\mathcal{I}}$;*
- *$\mathcal{I} \models \mathsf{Irr}(r)$, if $\{(x,x) \mid x \in \Delta^{\mathcal{I}}\} \cap r^{\mathcal{I}} = \emptyset$;*
- *$\mathcal{I} \models \mathsf{Sym}(r)$, if $(x,y) \in r^{\mathcal{I}} \rightarrow (y,x) \in r^{\mathcal{I}}$;*
- *$\mathcal{I} \models \mathsf{Asy}(r)$, if $(x,y) \in r^{\mathcal{I}} \rightarrow (y,x) \notin r^{\mathcal{I}}$;*
- *$\mathcal{I} \models \mathsf{Tra}(r)$, if $(x,y) \in r^{\mathcal{I}} \wedge (y,z) \in r^{\mathcal{I}} \rightarrow (x,z) \in r^{\mathcal{I}}$;*
- *$\mathcal{I} \models \mathsf{Dis}(r,s)$, if $r^{\mathcal{I}} \cap s^{\mathcal{I}} = \emptyset$;*

and therefore, $\mathcal{I} \models \mathcal{R}$, if $\mathcal{I} \models \mathcal{R}_h$ and $\mathcal{I} \models \phi$, for every role assertion $\phi \in \mathcal{R}_a$. For a TBox $\mathcal{T}$, $\mathcal{I} \models \mathcal{T}$, if for each GCI $C \sqsubseteq D$ in $\mathcal{T}$ the following holds for every interpretation $\mathcal{I}$: $\mathcal{I} \models C \sqsubseteq D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$; For axioms of an ABox $\mathcal{A}$:

- *$\mathcal{I} \models C(a)$, if $a^{\mathcal{I}} \in C^{\mathcal{I}}$;*
- *$\mathcal{I} \models r(a,b)$, if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$;*

- $\mathcal{I} \models \neg r(a,b)$, *if* $\mathcal{I} \not\models r(a,b)$;
- $\mathcal{I} \models a \approx b$, *if* $a^{\mathcal{I}} = b^{\mathcal{I}}$;
- $\mathcal{I} \models a \not\approx b$, *if* $\mathcal{I} \not\models a \approx b$;

Finally, $\mathcal{I} \models \mathcal{K}$, *if* $\mathcal{I} \models \alpha$, *for every axiom* $\alpha \in \mathcal{K}$. *A knowledge base* $\mathcal{K}$ *is called satisfiable or consistent, if it has a model, and it is called unsatisfiable or inconsistent or contradictory otherwise.*

2.1.3 First-Order Translation

Description logics are decidable first-order fragments, or at least can be translated to such. It is possible to provide translations for each axiom of some $\mathcal{SROIQ}$ knowledge base, into its corresponding first-order sentence. Table 2.1 depicts such a translation, which we will later reuse to obtain a grounded first-order translation.

$\tau(\mathcal{K})$	$\bigcup_{\alpha\in\mathcal{K}} \tau(\alpha)$
Axioms	
$\tau(C \sqsubseteq D)$	$\forall x.(\tau(C,x) \rightarrow \tau(D,x))$
$\tau(A(a))$	$A(a)$
$\tau(r(a,b) \,\vert\, a = b \,\vert\, a \neq b)$	$r(a,b) \,\vert\, a = b \,\vert\, a \neq b$
$\tau(s_1 \circ \cdots \circ s_n \sqsubseteq r)$	$\forall x, y_1, \ldots, y_n.(s_1(x,y_1) \wedge \cdots \wedge s_n(y_{n-1}, y_n) \rightarrow r(x,y_n))$
$\tau(\mathsf{Irr}(r) \,\vert\, \mathsf{Ref}(r))$	$\forall x,y.(r(x,y) \rightarrow x \neq y) \,\vert\, \forall x.(r(x,x))$
$\tau(\mathsf{Tra}(r))$	$\forall x,y,z.(r(x,y) \wedge r(y,z) \rightarrow r(x,z))$
$\tau(\mathsf{Sym}(r))$	$\forall x,y.(r(x,y) \rightarrow r(y,x))$
$\tau(\mathsf{Dis}(r,s))$	$\forall x,y.(r(x,y) \wedge s(x,y) \rightarrow \bot)$
$\tau(\mathsf{Asy}(r))$	$\forall x,y.(r(x,y) \wedge r(y,x) \rightarrow \bot)$
Concept Expressions	
$\tau(A,x)$	$A(x)$
$\tau(\neg C,x)$	$\neg\tau(C,x)$
$\tau(C \sqcup D,x)$	$\tau(C,x) \vee \tau(D,x)$
$\tau(C \sqcap D,x)$	$\tau(C,x) \wedge \tau(D,x)$
$\tau(\forall r.C,x)$	$\forall y.(r(x,y) \rightarrow \tau(C,y))$
$\tau(\exists r.C,x)$	$\exists y.(r(x,y) \wedge \tau(C,y))$
$\tau(\exists r.\mathit{Self},x)$	$r(x,x)$
$\tau(\{a\},x)$	$a = x$
$\tau(\geq n\ r.C,x)$	$\exists y_1,\ldots,y_n.(r(x,y_1) \wedge \ldots \wedge r(x,y_n) \wedge \tau(C,y_1) \wedge \ldots \wedge \tau(C,y_n) \wedge \bigwedge_{i\leq j} y_i \neq y_j)$
$\tau(\leq n\ r.C,x)$	$\neg\tau(\geq (n+1)\ r.C,x)$

Table 2.1: First-Order Translation.

2.1.4 Properties

Finite Model Property A property owned by many less expressive description logics is the so-called *finite model property*, which guarantees the existence of an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with finite domain $\Delta^{\mathcal{I}}$, such that $\mathcal{I} \models \mathcal{K}$, if $\mathcal{K}$ is satisfiable. $\mathcal{SROIQ}$ does not exhibit the finite model property; hence, it is possible to provide a knowledge base enforcing infinite models only. The following example depicts such axioms, it is based on a simpler version in [Cal96].

Example 2.9 (Infinite chain of guards)**.**

$$\begin{aligned} \mathsf{Guard} &\sqsubseteq \exists\, \mathsf{shields}.\mathsf{Guard} \sqcap \leq 1\, \mathsf{shields}^{-}.\mathsf{Guard} \\ \{\mathsf{firstguard}\} &\sqsubseteq \mathsf{Guard} \sqcap \leq 0\, \mathsf{shields}^{-}.\mathsf{Guard} \end{aligned}$$

A guard is someone who shields at least one guard and is itself shielded by at most one guard. Whereas a first-guard is a guard who is not shielded by another guard. Consequently, any model necessarily represents an infinite sequence of guards, each one shielding the next.

2.1.5 Reasoning Tasks

Given some knowledge base, as a matter of course particular inference questions arise. Beside the essential task of checking the knowledge base's satisfiability, its is desirable to provide further capabilities to derive new implicit knowledge. In the following, let $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$ be a $\mathcal{SROIQ}$ knowledge base, $C, D \in \mathbf{C}$ concept descriptions, and $a \in N_I^{dl}$ named individual.

Concept Satisfiability

C is satisfiable w.r.t. $\mathcal{K}$, iff $C^{\mathcal{I}} \neq \emptyset$ for some model $\mathcal{I}$ of $\mathcal{K}$. It is important to mention and understand here, that $C = \emptyset$ might very well hold in every model $\mathcal{I}$ of $\mathcal{K}$; i.e. satisfiable knowledge bases contain unsatisfiable concepts. On the other hand, a knowledge base where every atomic concept $A \in N_C^{dl}$ is satisfiable, is called *coherent* and is generally a desirable property as unsatisfiable concepts might bear modeling errors [Rud11].

ABox Consistency

$\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$ and $\mathcal{R}$, iff $\mathcal{A}$ has a model that is also a model of $\mathcal{T}$ and $\mathcal{R}$. Therefore, checking consistency of $\mathcal{A}$ w.r.t. $\mathcal{T}$ and $\mathcal{R}$ reduces to satisfiability testing of $\mathcal{K}$ itself.

Instance Retrieval

C has a as instance w.r.t. $\mathcal{K}$ ($\mathcal{K} \models C(a)$), iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{K}$. This is a rather strong statement since the membership must hold in every model, and thus there is no model $\mathcal{I}'$ of the ABox $\{\neg C(a)\}$, which is also a model for $\mathcal{K}$.

Subsumption Checking
C is subsumed by D w.r.t. $\mathcal{K}$ ($C \sqsubseteq D$), iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{K}$. In consequence, if also $D \sqsubseteq C$ w.r.t. $\mathcal{K}$ holds in every model $\mathcal{I}$ of $\mathcal{K}$, it can be concluded that $C \equiv D$ w.r.t. $\mathcal{K}$; i.e. $C^{\mathcal{I}} = D^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{K}$.

2.1.6 Tableaux Reasoning Services

Reasoning algorithms for expressive description logics nowadays widely implement the tableaux-based approach [BS01, BCM+07]. These algorithms aim at constructing a model for the given knowledge base [Rud11], while actually performing ABox consistency checking. Therefore, a tableaux structure is maintained, which to some extent can be seen as an ABox, successively updated under guidance of rule applications according to the model-theoretic semantics of the concept expression currently processed. For example, consider the complex individual assertion $(\exists r.(B \sqcup D))(a)$. According to the semantics (cf. 2.1.2), for any model $\mathcal{I}$ must hold, that $a^{\mathcal{I}} \in (\exists r.(B \sqcup D))^{\mathcal{I}}$, and consequently we can assume the existence of some b, such that $r(a, b)$ and $(B \sqcup D)(b)$ can be obtained and added to the current tableaux. Continuing, we can non-deterministically choose whether we add $B(b)$ or $D(b)$ to our model; referred to as OR-branching. This decision might later need to be rejected and backtracked to this very decision point, refuting the latest conclusions. This shall emphasize, that all tableaux rules, if applicable, make an expression more explicit by adding semantically more explicit expressions. Ultimately, in a successful execution, the algorithm terminates, since no rule is applicable anymore, or some terminating condition is met, yielding an ABox which itself induces a model of the knowledge base. Otherwise, inconsistency is detected if all branches lead to propositional clashes. We will not go into further details, and refer to [Sat07, BS01].

Optimizations With respect to $\mathcal{SROIQ}$, several sophisticated improvements have been proposed and implemented in HermiT, a fully OWL 2 DL compatible reasoner [MSH09, GHM+13]. Its underlying *hypertableaux* algorithm uses an extensive knowledge base preprocessing in order to exploit and eventually avoid many cases of non-determinism. To this end, axioms are translated into DL-clauses, allowing tableaux rules to be encoded differently, namely as forward rule application. For example, the DL-clause $A(X) \rightarrow B(X)$ directly allows to conclude $B(a)$ in the presence of $A(a)$.

2.2 Answer Set Programming

Answer set programming (ASP) is a modern approach towards declarative programming, where one focusses on modeling a problem purely by declarative means; i.e. not providing any form of algorithmic strategies on how to solve a problem. It originates from the field of logic programming, deductive databases, logic based knowledge representation and reasoning, constraint solving and satisfiability testing [GKKS12]. The idea of ASP is to represent a given computational problem by a logic program whose answer sets correspond to solutions [Lif02], an approach that suites well to tackle hard combinatorial search problems in NP, or $\mathsf{NP}^{\mathsf{NP}}$.

In the sequel we introduce syntax and semantics of the standardized language *ASP-Core-2* [CFG+13], a collaboratively developed standard in order to define a common ASP language, which so far has not existed. Merely, each implemented system was fledged with its own language features, all syntactically differently realized, such that comprehensive comparisons without changing a problem's encoding was impossible.

2.2.1 Syntax: ASP-Core-2

Definition 2.10 (Vocabulary)**.** *Let N_F^{lp}, N_R^{lp} and N_V^{lp} be three pairwise disjoint sets, where N_F^{lp} is a countable infinite set of function names, N_R^{lp} a countable infinite set of predicate names, and N_V^{lp} a countable infinite set of variables. Furthermore, a set of symbols for arithmetic connectives $\{+,-,*,/\} \subseteq N_F^{lp}$ and standard comparators $\{<,\leq,=,\neq,\geq,>\} \subseteq N_R^{lp}$ are considered.*

Definition 2.11 (Terms)**.** *The set of terms $\mathbf{T}$ is the smallest set, satisfying the following conditions:*

- *$X \in \mathbf{T}$, for all $X \in N_V^{lp}$;*
- *if $f/n \in N_F^{lp}$ and $\{t_1,\ldots,t_n\} \subseteq \mathbf{T}$, then $f(t_1,\ldots,t_n) \in \mathbf{T}$;*

A function of 0-arity is simply called constant, in particular we consider the integers $\mathbb{Z} \subseteq N_F^{lp}$ as special set of constants. For $\circ \in \{+,-,*,/\}$, terms t, $u \in \mathbf{T}$ the term $(t \circ u)$ is called arithmetic term. A term is said to be ground, if it not contains any variables.

Definition 2.12 (Atom, NaF-Literals)**.** *A predicate atom is of the form $p(t_1,\ldots,t_n)$, where $p \in N_R^{lp}$ is a predicate name, and $\{t_1,\ldots,t_n\} \subseteq \mathbf{T}$ are terms, for $n \geq 0$. Given a predicate atom q, q and $\neg q$ are classical atoms. For $\prec \in \{<,\leq,=,\neq,\geq,>\}$ and terms t, $u \in \mathbf{T}$ a built-in atom is of the form $t \prec u$. Built-in atoms α, as well as the expression α and $\mathsf{not}\ \alpha$ for a classical atom α are called naf-literals (nat = negation-as-failure).*

Definition 2.13 (Aggregate Literals)**.** *An aggregate element has the form $t_1,\ldots,t_m : l_1,\ldots,l_n$ where $t_1,\ldots,t_m$ are terms and $l_1,\ldots,l_n$ are naf-literals for $m \geq 0$ and $n \geq 0$. An aggregate atom then is of the form $\#aggr\{e_1;\ldots;e_n\} \prec u$, where $\#aggr \in \{\#\mathsf{count},\#\mathsf{sum},\#\mathsf{min},\#\mathsf{max}\}$ is an aggregate function name, $e_1,\ldots,e_n$ are aggregate elements for $n \geq 0$, $\prec \in \{<,\leq,=,\neq,\geq,>\}$ is an aggregate relation and u is a term.*

Definition 2.14 (Rules, Programs)**.** *An ASP-2-Core program* Π*, is a finite set of rules, each of the form*

$$h_1 \mid \ldots \mid h_m \coloneq b_1, \ldots, b_k, \ \mathsf{not}\ b_{k+1}, \ldots, \ \mathsf{not}\ b_n.$$

where $h_1 \mid \ldots \mid h_m$ *is a disjunction of classical atoms* $h_1, \ldots, h_m$*, and* $b_1, \ldots, b_n$ *are literals conjunctively connected for* $n \geq k \geq 0$ *and* $m \geq 0$*. A rule is ground if it contains no variables, and in consequence, program* Π *is ground if every rule* $r \in \Pi$ *is ground.*

2.2.2 Semantics: ASP-Core-2

Definition 2.15 (Herbrand Interpretation)**.** *Given a program* Π*, the Herbrand universe of* Π*, denoted by* $\mathcal{U}_\Pi$*, consists of all integers and ground terms constructible from constants and functors appearing in* Π*. The Herbrand base of* Π*, denoted by* $\mathcal{B}_\Pi$*, is the set of all ground classical atoms that can be built by combining predicate names appearing in* Π *with terms from* $\mathcal{U}_\Pi$ *as arguments. Then, a Herbrand interpretation* $\mathcal{I}$ *for* Π *is a consistent subset of* $\mathcal{B}_\Pi$*; that is,* $\{q, \neg q\} \not\subseteq \mathcal{I}$ *must hold for each predicate atom* $q \in B_\Pi$*.*

Definition 2.16 (Ground Instantiation)**.** *A substitution* σ *is a mapping from the set* N_V^{lp} *of variables to the Herbrand universe* $\mathcal{U}_\Pi$ *of a given program* Π*. For some object* O *(term, classical atom, literal, rule, etc.), the object* $O\sigma$ *is obtained by replacing each occurrence of a variable* $X \in N_V^{lp}$ *by* $\sigma(X)$ *in* O*. A variable is global in a rule* r*, if it appears outside of aggregate elements in* r*. A substitution from the set of global variables in* r *is a global substitution for* r*; a substitution from the set of variables in an aggregate element* e *is a local substitution for* e*. A global substitution* σ *for* r *or* e *is well-formed, if the arithmetic evaluation performed in the standard way of any arithmetic subterm appearing outside of aggregate elements in* $r\sigma$ *or* $e\sigma$ *is well-defined. Given a collection* $\{e_1; \ldots; e_n\}$ *of aggregate elements, the instantiation of* $\{e_1; \ldots; e_n\}$ *is the following set of aggregate elements:*

$$inst(\{e_1; \ldots; e_n\}) \coloneq \bigcup_{1 \leq i \leq n} \{e_i\sigma \mid \sigma \text{ is a well-formed substitution for } e_i\}$$

A ground instance of a term, classical atom, naf-literal or rule r *is obtained in two steps:*

1. *a well-formed global substitution* σ *for* r *is applied to* r*;*
2. *for every aggregate atom* $\#aggr\{e_1; \ldots; e_n\} \prec u$ *appearing in* $r\sigma$*,* $\{e_1; \ldots; e_n\}$ *is replaced by* $inst(\{e_1; \ldots; e_n\})$*.*

The arithmetic evaluation of a ground instance r *of some term, classical atom, naf-literal or rule is obtained by replacing any maximal arithmetic subterm appearing in* r *by its integer value, which is calculated in the standard way. The ground instantiation of a program* Π*, denoted by* $grnd(\Pi)$*, is the set of arithmetically evaluated ground instances of rules in* Π*.*

Definition 2.17 (Term ordering, Naf-Literal Satisfiability)**.** *A ground classical atom $\alpha \in \mathcal{B}_\Pi$ is true w.r.t. a consistent interpretation $\mathcal{I} \subseteq \mathcal{B}_\Pi$, if $\alpha \in \mathcal{I}$. Let t and u be arithmetically evaluated ground terms. Determining whether a built-in atom $t \prec u$, with $\prec \in \{<, \leq, =, \neq, \geq, >\}$, holds, we rely on a total order $\preceq$ on terms in $\mathcal{U}_\Pi$ defined as follows:*

- *$t \preceq u$ for integers t and u, if $t \leq u$;*
- *$t \preceq u$ for any integer t and any symbolic constant u;*
- *$t \preceq u$ for symbolic constants t and u, if t is lexicographically smaller than or equal to u;*
- *$t \preceq u$ for any symbolic constant t and any string constant u;*
- *$t \preceq u$ for string constants t and u, if t is lexicographically smaller than or equal to u;*
- *$t \preceq u$ for any string constant t and any functional term u;*
- *$t \preceq u$ for functional terms $t = f(t_1, \ldots, t_m)$ and $u = g(u_1, \ldots, u_n)$ if*
  - *$m < n$,*
  - *$m \leq n$ and $g \not\preceq f$, or*
  - *$m \leq n$, $f \preceq g$ and, for any $1 \leq j \leq m$ sucht that $t_j \not\preceq u_j$, there is some $1 \leq i < j$ sucht that $u_i \not\preceq t_i$.*

Consequently, $t \prec u$ is true w.r.t. $\mathcal{I}$ if $t \preceq u$ for $\prec =$ " $\leq$ "; $u \preceq t$ for $\prec =$ " $\geq$ "; $t \prec u$ and $u \not\preceq t$ for $\prec =$ " $<$ "; $u \preceq t$ and $t \not\preceq u$ for $\prec =$ " $>$ "; $t \preceq u$ and $u \preceq t$ for $\prec =$ " $=$ "; $t \not\preceq u$ or $u \not\preceq t$ for $\prec =$ " $\neq$ ". A positive ground naf-literal α is true w.r.t. $\mathcal{I}$ if α is a classical or built-in atom that is true w.r.t. $\mathcal{I}$; otherwise, α is false w.r.t. $\mathcal{I}$. A negative ground naf-literal not α *is true (or false) w.r.t. $\mathcal{I}$ if α is false (or true) w.r.t. $\mathcal{I}$.*

Definition 2.18 (Aggregate Literal Satisfiability)**.** *An aggregate function is a mapping from set of tuples of terms to terms. The aggregate functions associated with aggregate function names introduced in Definition 2.13 map a set $\mathbf{T_T}$ of tuples of ground terms to a ground term as follows:*

$$\begin{aligned}
\#\mathsf{count}(\mathbf{T_T}) &:= |\mathbf{T_T}| && (2.6)\\
\#\mathsf{sum}(\mathbf{T_T}) &:= \Sigma_{(t_1,\ldots,t_n)\in\mathbf{T_T}, t_1\in\mathbb{Z}} t_1 && (2.7)\\
\#\mathsf{max}(\mathbf{T_T}) &:= max\{t_1 |\ (t_1, \ldots, t_n) \in \mathbf{T_T}\} && (2.8)\\
\#\mathsf{min}(\mathbf{T_T}) &:= min\{t_1 |\ (t_1, \ldots, t_n) \in \mathbf{T_T}\} && (2.9)
\end{aligned}$$

The terms selected by $\#\mathsf{max}(\mathbf{T_T})$ and $\#\mathsf{min}(\mathbf{T_T})$ are determined relative to the total order $\preceq$ on terms in $\mathcal{U}_\Pi$; in the special case of an empty set, i.e. $\mathbf{T_T} = \emptyset$, we adopt the convention that $\#\mathsf{max}(\emptyset) \preceq u$ and $u \preceq \#\mathsf{min}(\emptyset)$ for every term $u \in \mathcal{U}_\Pi$. An expression $\#aggr(\mathbf{T_T}) \prec u$ is true (or false) for $\#aggr \in \{$"$\#\mathsf{count}$", "$\#\mathsf{sum}$", "$\#\mathsf{min}$", "$\#\mathsf{max}$"$\}$, and aggregate relation $\prec \in \{<, \leq, =, \neq, \geq, >\}$ and a ground term u, if $\#aggr(\mathbf{T_T}) \prec u$ is true (or false) according to the corresponding definition for built-in atoms given in

Definition 2.17. An interpretation $\mathcal{I} \subseteq \mathcal{B}_\Pi$ maps a collection E of aggregate elements to the following set of tuples of ground terms:

$$eval(E,\mathcal{I}) := \{ (t_1,\ldots,t_m) \mid t_1,\ldots,t_m : l_1,\ldots,l_n \text{ occurs in } E \text{ and } l_1,\ldots,l_n \text{ are true w.r.t. } \mathcal{I} \}$$

A positive aggregate literal $\alpha = \#aggr\{e_1;\ldots;e_n\} \prec u$ is true (or false) w.r.t. $\mathcal{I}$ if $\#aggr(eval(\{e_1;\ldots;e_n\},\mathcal{I})) \prec u$ is true (or false) w.r.t. $\mathcal{I}$; not α is true (or false) w.r.t. $\mathcal{I}$, if α is false (or true) w.r.t. $\mathcal{I}$.

Definition 2.19 (Answer Sets)**.** *Given a program Π and (consistent) interpretation $\mathcal{I} \subseteq \mathcal{B}_\Pi$, a rule $r \in grnd(\Pi)$ is satisfied w.r.t. $\mathcal{I}$ if some $h \in H(r)$ is true w.r.t. $\mathcal{I}$ when $b \in B(r)$ is true w.r.t. $\mathcal{I}$; $\mathcal{I}$ is a model of Π if every rule in $grnd(\Pi)$ is satisfied w.r.t. $\mathcal{I}$. The reduct of Π w.r.t. $\mathcal{I}$, denoted by $\Pi^\mathcal{I}$, consists of all rules $r \in grnd(\Pi)$, such that $B(r)$ is true w.r.t. $\mathcal{I}$; $\mathcal{I}$ is an answer set of Π if $\mathcal{I}$ is a $\subseteq$-minimal model of $\Pi^\mathcal{I}$. That is, an answer set $\mathcal{I}$ of Π is a model of Π such that no proper subset of $\mathcal{I}$ is a model of $\Pi^\mathcal{I}$. The semantics of Π is given by the collection of its answer sets, denoted by $\mathsf{AS}(\Pi)$.*

2.2.3 Modeling in ASP

The term *programming* is potentially misleading, given that, one actually focusses on *modeling* a problem purely by declarative means; i.e. not providing any form of algorithmic aspects on how to solve a problem.

Guess-and-Check Paradigm Commonly referred to as *Guess-and-Check* paradigm, or *Generate-and-Test* [GKKS12], the general pattern of most answer set programs consist of one part, *generating* potential solution candidates, and thereby defining the search space, and another part *checks* each potential solution whether it indeed represents a solution of the encoded problem. This also depicts the nature of NP; i.e. non-deterministic guessing of prospective solutions, and subsequent checking, realizable in polynomial time.

2.3 Summary

2.3.1 Description Logics vs. Answer Set Programs

Both knowledge representation and reasoning approaches are quite of contrary nature. Whereas description logics classically adhere to an *open-world assumption* (OWA), answer set programs operate under the *closed-world assumption* (CWA). That is, the truth value of missing information in some DL knowledge base, is unknown. On the other hand, the semantic *naf*-operator (*default negation*) admits to test on the existence or non-existence of some fact, and if so, conclude the contrary. We have just demonstrated how this is utilized for guessing. The answer set semantics also imposes *non-monotic* behavior, something we can also not find in DLs by default. As first-order fragment, DLs naturally operate in monotonic waters.

Carefully considered, in both approaches reasoning algorithms conduct a search for a model. Whereas in DLs, tableaux algorithms try to successively construct some model, answer set solvers use elimination techniques and thus perform some sort of directed search, using techniques from modern SAT solvers like *conflict-driven clause learning* or the combination with no-good-learning, yielding *conflict-driven no-good learning* [GKKS12].

3 Bounded Model Reasoning

We coin the notion of a *bounded model* and discuss related notions in order to place our semantics into proper contexts. We then further look into reasoning with respect to bounded models and illustrate that standard reasoning tasks are straight forward applicable, whereas our motivation exhibits new reasoning tasks. It is shown that *bounded model reasoning* in $\mathcal{SROIQ}$ is NP-complete, which might leave the impression that this is an easier task for existing tableaux based reasoners performing exceptionally well for standard semantics. However, we exemplify the problem of employing existing reasoners to reason with respect to bounded models, leading us to our approach described in the subsequent chapter.

3.1 Intuition

Usually there are no restrictions imposed on the size of interpretations when reasoning in description logics. In order to check satisfiability, for example, one is only interested whether some model exists rather than searching for a specific one. Nevertheless, in many real applications, the domain of interest is actually finite, such that already quite a lot of effort has been put into reasoning with interpretations where the eventual cardinality is unknown but finite, known as *finite model reasoning* [LST05, Cal96, Ros08].

In the sequel we rather look into finite models of *a priori known cardinality*, in particular where the cardinality is induced by all named individuals occurring in the knowledge base of interest. We refer to them as *bounded models* and argue that in many applications this admittedly minor modification of the standard DL semantics represents a more intuitive definition of what is considered and expected as *model* of some knowledge base.

3.2 Semantics

Definition 3.1 (Individual Bounded Interpretation)**.** *Let $\mathcal{K}$ be a $\mathcal{SROIQ}$ knowledge base. An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is said to be individual bounded w.r.t. $\mathcal{K}$, if all of the following holds:*

(i) $\Delta^{\mathcal{I}} = \{a \mid a \in N_I^{dl}(\mathcal{K})\} \neq \emptyset$,

(ii) for each individual $a \in N_I^{dl}(\mathcal{K})$, $a^{\mathcal{I}} = a$,

(iii) for each concept name $A \in N_C^{dl}(\mathcal{K})$, $A^{\mathcal{I}} \subseteq \{a \mid a \in N_I^{dl}(\mathcal{K})\}$, *and*

(iv) for each role name $r \in N_R^{dl}(\mathcal{K})$, $r^{\mathcal{I}} \subseteq \{a \mid a \in N_I^{dl}(\mathcal{K})\} \times \{a \mid a \in N_I^{dl}(\mathcal{K})\}$.

Accordingly, we call an interpretation $\mathcal{I}$ individual bounded model of $\mathcal{K}$, if $\mathcal{I}$ is a individual bounded interpretation and $\mathcal{I} \models \mathcal{K}$ holds; i.e. $\mathcal{I} \models \alpha$, for every axiom $\alpha \in \mathcal{K}$.

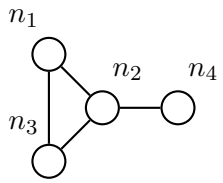
Note that (iii) and (iv) could of course be denoted with $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, respectively, as it is in the standard semantics. However, Definition 3.1 shall explicitly stress the individual bounded extensions of concept and roles names. The notion of *logical consequence* is defined as usual: an axiom α is a logical consequence of a given knowledge base $\mathcal{K}$, if and only if α is true in all bounded models of $\mathcal{K}$; in fact, the semantic notions introduced in Section 2.1.2 remain intact, as well as reasoning w.r.t. bounded models can be straight-forward adapted from the reasoning tasks given in Section 2.1.5, which we will from now on refer to as *standard reasoning tasks* and discuss in more detail in Section 3.3. First consider the following example.

Example 3.2 (Graph Coloring)**.** *The problem of coloring the nodes of some given graph structure with only three colors, such that two nodes connected via an edge are not allowed to share the same color. We can formalize the problem intuitively in a uniform way, as $\mathcal{K}_{3Col} = (\mathcal{R}_{3Col}, \mathcal{T}_{3Col}, \emptyset)$, where $\mathcal{R}_{3Col}$, $\mathcal{T}_{3Col}$ are defined as follows:*

$$\begin{aligned}
\mathcal{T}_{3Col} = \{ \quad \mathsf{Node} &\sqsubseteq \mathsf{RNode} \sqcup \mathsf{BNode} \sqcup \mathsf{GNode}, && (3.1)\\
\mathsf{RNode} &\sqsubseteq \forall\, \mathsf{edge}.(\mathsf{BNode} \sqcup \mathsf{GNode}), && (3.2)\\
\mathsf{GNode} &\sqsubseteq \forall\, \mathsf{edge}.(\mathsf{RNode} \sqcup \mathsf{BNode}), && (3.3)\\
\mathsf{BNode} &\sqsubseteq \forall\, \mathsf{edge}.(\mathsf{GNode} \sqcup \mathsf{RNode}) && (3.4)\\
\mathsf{BNode} \sqcap \mathsf{RNode} &\sqsubseteq \bot && (3.5)\\
\mathsf{BNode} \sqcap \mathsf{GNode} &\sqsubseteq \bot && (3.6)\\
\mathsf{RNode} \sqcap \mathsf{GNode} &\sqsubseteq \bot \quad \} && (3.7)
\end{aligned}$$

$$\mathcal{R}_{3Col} = \{ \ \mathsf{Irr}(\mathsf{edge}),\ \mathsf{Sym}(\mathsf{edge}) \ \} \qquad (3.8)$$

Axiom (3.1) expresses that an instance of Node *is a red* (RNode)*, blue* (BNode) *or green node* (GNode)*. Whereas the axioms (3.2–3.4) state, that some red node has only edges to blue or green nodes, and likewise for the other colors. Axioms (3.5–3.7) ensure pairwise disjointness of the colored nodes. We use the role assertions* Irr(edge) *to disallow a node to have an edge to itself, and* Sym(edge) *realizing an undirected graph. With $\mathcal{K}_{3Col}$ we are not able to provide any bounded model since it is not possible to identity a domain due to missing individuals. Therefore, let $\mathcal{A}_{sim}$ denote the following simple input graph:*[1]



$$\mathcal{A}_{sim} = \{ \ \mathsf{Node}(n_1; n_2; n_3; n_4), \mathsf{edge}(n_1, n_2), \mathsf{edge}(n_1, n_3), \mathsf{edge}(n_2, n_3), \mathsf{edge}(n_2, n_4) \ \}$$

[1] We use a short-hand notation for assigning several individuals to the same concept via ";". E.g. $\mathsf{Node}(n_1; n_2)$ is a syntactic shorthand notation for $\mathsf{Node}(n_1)$ and $\mathsf{Node}(n_2)$.

Henceforth, we are able to provide bounded interpretations over the domain $\Delta_{3Col} = \{n_1, \ldots, n_4\}$*, and for example verify that the interpretation* $\mathcal{I}_1 = (\Delta_{3Col}, \cdot^{\mathcal{I}_1})$ *with*

$$\mathsf{RNode}^{\mathcal{I}_1} = \{n_1, n_4\}, \mathsf{BNode}^{\mathcal{I}_1} = \{n_3\}, \mathsf{GNode}^{\mathcal{I}_1} = \{n_2\},$$

and obligatory extensions to satisfy $\mathcal{A}_{sim}$*, is indeed a bounded model for* $\mathcal{K}_{3Col} = (\mathcal{R}_{3Col}, \mathcal{T}_{3Col}, \mathcal{A}_{sim})$ *not violating any constraints imposed by the graph coloring problem.*

Example 3.2 illustrates the graph coloring problem formalized by means of few DL axioms and points out the intuitive idea of bounded models, representing a valid coloring of some given graph. On the contrary, example 3.3 depicts an unsatisfiable knowledge base w.r.t. bounded models. Apparently, knowledge bases enforcing an infinite domain are unsatisfiable w.r.t. bounded models, which is a direct consequence of the bounded domain definition.

Example 3.3 (Unsatisfiable Knowledge Base)**.** *We reconsider Example 2.9:*

$$\begin{aligned} \mathsf{Guard} &\sqsubseteq \exists\, \mathsf{shields}.\mathsf{Guard} \sqcap \leq 1\, \mathsf{shields}^{-}.\mathsf{Guard} \\ \{\mathsf{firstguard}\} &\sqsubseteq \mathsf{Guard} \sqcap \leq 0\, \mathsf{shields}^{-}.\mathsf{Guard} \end{aligned}$$

With the induced bounded domain $\Delta = \{\mathsf{firstguard}\}$ *it is not possible to construct an interpretation* $\mathcal{I}$ *satisfying both axioms, such that* $\mathsf{firstguard}^{\mathcal{I}} \in (\leq 0\, \mathsf{shields}^{-}.\mathsf{Guard})^{\mathcal{I}}$ *and* $\mathsf{firstguard}^{\mathcal{I}} \in (\leq 1\, \mathsf{shields}^{-}.\mathsf{Guard})$*.*

3.2.1 Related Notions

To the best of our knowledge, the term *bounded model* is not preallocated in the field of DL or the Semantic Web. There is, nevertheless, a closely related notion which we want to introduce briefly. That is, in the context of tableaux based reasoners, the so-called *canonical interpretation* is used to show that any complete and open ABox $\mathcal{A}$, obtained by applying tableaux rules, is indeed a (*canonical*) model of $\mathcal{A}$.

Definition 3.4 (Canonical Interpretation)**.** *Let* $\mathcal{A}$ *be an ABox. The canonical interpretation* $\mathcal{I}_{\mathcal{A}} = (\Delta^{\mathcal{I}_{\mathcal{A}}}, \cdot^{\mathcal{I}_{\mathcal{A}}})$ *induced by* $\mathcal{A}$*, is defined as follows:*

- $\Delta^{\mathcal{I}_{\mathcal{A}}} := \{a \in N_I^{dl} \mid a \text{ occurs in } \mathcal{A}\}$,
- *for all* $a \in N_I^{dl}$ *occurring in* $\mathcal{A}$, $a^{\mathcal{I}_{\mathcal{A}}} := a$,
- *for all concept names* $A \in N_C^{dl}$, $A^{\mathcal{I}_{\mathcal{A}}} := \{a \mid A(a) \in \mathcal{A}\}$, *and*
- *for all role names* $r \in N_R^{dl}$, $r^{\mathcal{I}_{\mathcal{A}}} := \{(a, b) \mid r(a, b) \in \mathcal{A}\}$.

The domain of a canonical interpretation consists of all individuals occurring in $\mathcal{A}$, and each of these individuals is mapped to itself. The main idea is that the extension of concepts and roles is obtained from concept assertions and role assertions present in $\mathcal{A}$. In a way this corresponds to Herbrand's idea, and his notion of *Herbrand interpretations*

which is widely known for first-order theories[2]. Within the first-order translation $\tau(\mathcal{K})$ for some knowledge base $\mathcal{K}$, shown in Chapter 2.1.3, individuals ultimately translate to constants and in fact represent the only function symbols in the signature of the obtained translation. Hence, the set of constants occurring in $\tau(\mathcal{K})$ constitute the Herbrand universe $\mathcal{U}_{\tau(\mathcal{K})}$, such that the Herbrand base $\mathcal{B}_{\tau(\mathcal{K})}$ is the set of all ground atoms constructible by combining predicate names and elements of $\mathcal{U}_{\tau(\mathcal{K})}$ as arguments. The relation to bounded interpretations is pervasive, which we will further elaborate in the next section.

3.2.2 Alternative Characterization

Inspired by canonical interpretations and the direct interrelation to Herbrand interpretations, it seems likely to extend the notion in an analogous way, by means of defining an interpretation $\mathcal{I}_{\mathbf{B}}$ induced by some specific set of assertions $\mathbf{B}$ constructible over the signature of $\mathcal{K}$. We require $\mathbf{B}$ to contain a (negative or positive) assertion for each individual and concept name combination, respectively for each two individuals and role name. The following example illustrates the idea.

Example 3.5. *Regarding the graph coloring example, consider the following different sets of assertions:*

$$\begin{aligned}
\mathbf{B}_1 \;=\; \{\; & \mathsf{Node}(n_1;n_2;n_3;n_4), \mathsf{edge}(n_1,n_2), \mathsf{edge}(n_1,n_3), \mathsf{edge}(n_2,n_3), \mathsf{edge}(n_2,n_4),\\
& \neg\mathsf{edge}(n_1,n_4), \neg\mathsf{edge}(n_3,n_4), \mathsf{RNode}(n_1;n_4), \neg\mathsf{RNode}(n_2;n_3), \mathsf{BNode}(n_3),\\
& \neg\mathsf{BNode}(n_1;n_2;n_4), \mathsf{GNode}(n_2), \neg\mathsf{GNode}(n_1;n_3;n4) \;\}\\
\mathbf{B}_2 \;=\; \{\; & \mathsf{Node}(n_1;n_2;n_3;n_4), \neg\mathsf{edge}(n_1,n_2), \mathsf{edge}(n_1,n_3), \mathsf{edge}(n_2,n_3), \mathsf{edge}(n_2,n_4),\\
& \neg\mathsf{edge}(n_1,n_4), \neg\mathsf{edge}(n_3,n_4), \mathsf{RNode}(n_1;n_4), \neg\mathsf{RNode}(n_2;n_3), \mathsf{BNode}(n_3),\\
& \neg\mathsf{BNode}(n_1;n_2;n_4), \mathsf{GNode}(n_2), \neg\mathsf{GNode}(n_1;n_3;n_4) \;\}\\
\mathbf{B}_3 \;=\; \{\; & \mathsf{Node}(n_1;n_2;n_3;n_4), \mathsf{edge}(n_1,n_2), \mathsf{edge}(n_1,n_3), \mathsf{edge}(n_2,n_3), \mathsf{edge}(n_2,n_4),\\
& \neg\mathsf{edge}(n_1,n_4), \neg\mathsf{edge}(n_3,n_4), \neg\mathsf{RNode}(n_1;n_2;n_3;n_4),\\
& \mathsf{BNode}(n_3), \neg\mathsf{BNode}(n_1;n_2;n_4), \mathsf{GNode}(n_2), \neg\mathsf{GNode}(n_1;n_3;n_4) \;\}
\end{aligned}$$

The interpretation induced by $\mathbf{B}_1$ *defines a model for* $\mathcal{K}_{3Col} = (\mathcal{R}_{3Col}, \mathcal{T}_{3Col}, \mathcal{A}_{sim})$*, whereas for* $\mathbf{B}_2$ *and* $\mathbf{B}_3$ *this is not the case, as* $\neg\mathsf{edge}(n_1,n_2) \in \mathbf{B}_2$ *conflicts with* $\mathsf{edge}(n_1,n_2) \in \mathcal{A}_{sim}$*, and* $\mathbf{B}_3$ *simply does not permit any coloring not violating the constraints.*

This yields a different characterization of bounded models, which we temporarily will refer to as *grounded model* and is formalized in the following definition.

Definition 3.6 (Grounded Interpretation)**.** *Let* $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$ *be a* $\mathcal{SROIQ}$ *knowledge base and* $\mathcal{B}_{\mathcal{K}}$ *be the set of all sets* $\mathbf{B}$*, where each* $\mathbf{B}$ *is the smallest set constructed as follows:*

[2] We introduced the notion of *Herbrand base*, *Herbrand universe* and *Herbrand interpretation* already in Chapter 2.2.2.

- *for every concept name* $A \in N_C^{dl}(\mathcal{K})$ *and individual* $a \in N_I^{dl}(\mathcal{K})$*, either* $A(a) \in \mathbf{B}$ *or* $\neg A(a) \in \mathbf{B}$*, and*
- *for every role name* $r \in N_R^{dl}(\mathcal{K})$ *and individuals* $a, b \in N_I^{dl}(\mathcal{K})$*, either* $r(a,b) \in \mathbf{B}$ *or* $\neg r(a,b) \in \mathbf{B}$*.*

Then $\mathcal{I}_\mathbf{B} = (\Delta^{\mathcal{I}_\mathbf{B}}, \cdot^{\mathcal{I}_\mathbf{B}})$ *is an interpretation induced by some* $\mathbf{B} \in \mathcal{B}_\mathcal{K}$ *as follows:*

- $\Delta^{\mathcal{I}_\mathbf{B}} := \{a \in N_I^{dl} \mid a \text{ occurs in } \mathbf{B}\}$,
- *for all* $a \in N_I^{dl}$ *occurring in* $\mathbf{B}$*,* $a^{\mathcal{I}_\mathbf{B}} := a$*,*
- *for all concept names* $A \in N_C^{dl}$*,* $A^{\mathcal{I}_\mathbf{B}} := \{a \mid A(a) \in \mathbf{B}\}$*, and*
- *for all role names* $r \in N_R^{dl}$*,* $r^{\mathcal{I}_\mathbf{B}} := \{(a,b) \mid r(a,b) \in \mathbf{B}\}$*.*

$\mathcal{I}_\mathbf{B}$ *is called grounded model of* $\mathcal{K}$*, if* $\mathcal{I}_\mathbf{B} \models \alpha$ *holds for every axiom* $\alpha \in \mathcal{K}$*.*

Note that $|\mathcal{B}_\mathcal{K}| = 2^{nm} + 2^{n^2 o}$, with $n = N_I^{dl}(\mathcal{K})$, $m = N_C^{dl}(\mathcal{K})$ and $o = N_R^{dl}(\mathcal{K})$. $\mathcal{I}_\mathbf{B}$ is defined in the same way as the canonical interpretation, and we claim now that any bounded model uniquely determines a grounded model, as well as any grounded model actually is a bounded model. The following lemmas underpin these findings.

Lemma 3.7. *Let* $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$ *be a* $\mathcal{SROIQ}$ *knowledge base. Then for any bounded interpretation* $\mathcal{I}$ *w.r.t.* $\mathcal{K}$ *it holds:* $\mathcal{I} \models \mathcal{K} \implies \mathcal{I}_\mathbf{B} \models \mathcal{K}$*, where* $\mathbf{B}$ *is obtained from* $\mathcal{I}$ *as follows:*

- $A(a) \in \mathbf{B}$ *if* $a^\mathcal{I} \in A^\mathcal{I}$*, for every* $A \in N_C^{dl}(\mathcal{K})$ *and* $a \in N_I^{dl}(\mathcal{K})$*,*
- $\neg A(a) \in \mathbf{B}$ *if* $a^\mathcal{I} \notin A^\mathcal{I}$*, for every* $A \in N_C^{dl}(\mathcal{K})$ *and* $a \in N_I^{dl}(\mathcal{K})$*,*
- $r(a,b) \in \mathbf{B}$ *if* $(a^\mathcal{I}, b^\mathcal{I}) \in r^\mathcal{I}$*, for every* $r \in N_R^{dl}(\mathcal{K})$ *and* $a, b \in N_I^{dl}(\mathcal{K})$*,*
- $\neg r(a,b) \in \mathbf{B}$ *if* $(a^\mathcal{I}, b^\mathcal{I}) \notin r^\mathcal{I}$*, for every* $r \in N_R^{dl}(\mathcal{K})$ *and* $a, b \in N_I^{dl}(\mathcal{K})$*.*

Proof. Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be some bounded interpretation for $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$, such that $\mathcal{I} \models \mathcal{K}$ holds, and $\mathcal{I}_\mathbf{B} = (\Delta^{\mathcal{I}_\mathbf{B}}, \cdot^{\mathcal{I}_\mathbf{B}})$ be the interpretation induced by $\mathbf{B}$ obtained from $\mathcal{I}$. It is easy to see that $\Delta^\mathcal{I} = \Delta^{\mathcal{I}_\mathbf{B}}$, such that $\top^\mathcal{I} = \top^{\mathcal{I}_\mathbf{B}}$, as well as $\bot^\mathcal{I} = \emptyset = \bot^{\mathcal{I}_\mathbf{B}}$. Moreover for every $a \in \Delta^\mathcal{I}$ we have that $a^\mathcal{I} = a = a^{\mathcal{I}_\mathbf{B}}$ by definition of $\mathcal{I}_\mathbf{B}$. We first show for any $a \in C^\mathcal{I} \implies a \in C^{\mathcal{I}_\mathbf{B}}$, by induction on the structure of concept expression C. Let

$\underline{C = A \in N_C^{dl}}$: then for any $a^\mathcal{I} \in A^\mathcal{I}$ we find $A(a) \in \mathbf{B}$ and therefore $a \in A^{\mathcal{I}_\mathbf{B}}$.

$\underline{C = \neg A \in N_C^{dl}}$: then for each $a^\mathcal{I} \notin A^\mathcal{I}$ we find $\neg A(a) \in \mathbf{B}$ and consequently $a \notin A^{\mathcal{I}_\mathbf{B}}$ by definition of $\mathcal{I}_\mathbf{B}$.

$\underline{C = C_1 \sqcup C_2}$: then for any $a^\mathcal{I} \in (C_1 \sqcup C_2)^\mathcal{I}$, $a^\mathcal{I} \in C_1^\mathcal{I}$ or $a^\mathcal{I} \in C_2^\mathcal{I}$. By induction, $a \in C_1^{\mathcal{I}_\mathbf{B}}$ if $a^\mathcal{I} \in C_1^\mathcal{I}$, or $a \in C_2^{\mathcal{I}_\mathbf{B}}$ if $a^\mathcal{I} \in C_2^\mathcal{I}$. And consequently $a \in (C_1 \sqcup C_2)^{\mathcal{I}_\mathbf{B}}$.

$\underline{C = \exists r.D}$: then for any $a^{\mathcal{I}} \in (\exists r.D)^{\mathcal{I}}$ there exists $b \in \Delta^{\mathcal{I}}$, such that $(a,b) \in r^{\mathcal{I}}$ and $b \in D^{\mathcal{I}}$. For $(a,b) \in r^{\mathcal{I}}$ we find $r(a,b) \in \mathbf{B}$ and therefore $(a,b) \in r^{\mathcal{I}_\mathbf{B}}$, as well as $b \in D^{\mathcal{I}}$ implies $D(b) \in \mathbf{B}$ for which we obtain $b \in D^{\mathcal{I}_\mathbf{B}}$ and therefore $a \in (\exists r.D)^{\mathcal{I}_\mathbf{B}}$.

$\underline{C = \geq n\, r.D}$: then for any $a^{\mathcal{I}} \in (\geq n\, r.D)^{\mathcal{I}}$ there are at least n $b \in \Delta^{\mathcal{I}}$, such that $(a,b) \in r^{\mathcal{I}}$ and $b \in D^{\mathcal{I}}$. From each of those $(a,b) \in r^{\mathcal{I}}$ we have $r(a,b) \in \mathbf{B}$ and therefore $(a,b) \in r^{\mathcal{I}_\mathbf{B}}$, and from $b \in D^{\mathcal{I}}$ we obtain $b \in D^{\mathcal{I}_\mathbf{B}}$ by induction. And therefore $a \in (\geq n\, r.D)^{\mathcal{I}_\mathbf{B}}$.

$\underline{C = \exists r.Self}$: then for any $a^{\mathcal{I}} \in (\exists r.Self)^{\mathcal{I}}$ also $(a,a) \in r^{\mathcal{I}}$, and hence $r(a,a) \in \mathbf{B}$ such that $(a,a) \in r^{\mathcal{I}_\mathbf{B}}$ and therefore $a \in (\exists r.Self)^{\mathcal{I}_\mathbf{B}}$.

All other constructors are treated analogously.

Then $\mathcal{I} \models \alpha$ for every axiom $\alpha \in \mathcal{K}$, and it remains to show that $\mathcal{I}_\mathbf{B} \models \alpha$. Let

$\underline{\alpha \in \mathcal{A}}$: We distinguish assertions on complex concept expression $C \in \mathbf{C}$, and atomic role name $r \in N_R^{dl}$.

$\underline{\alpha = r(a,b)}$: then $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$ and therefore $r(a,b) \in \mathbf{B}$, and consequently $(a,b) \in r^{\mathcal{I}_\mathbf{B}}$ by definition of $\mathcal{I}_\mathbf{B}$.

$\underline{\alpha = \neg r(a,b)}$: then $\mathcal{I} \not\models r(a,b)$, such that $(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin r^{\mathcal{I}}$ and therefore $\neg r(a,b) \in \mathbf{B}$, and consequently $(a,b) \notin r^{\mathcal{I}_\mathbf{B}}$ by definition of $\mathcal{I}_\mathbf{B}$.

$\underline{\alpha = C(a)}$: then for any $C \in \mathbf{C}$ we have show that $a^{\mathcal{I}} \in C^{\mathcal{I}} \implies a^{\mathcal{I}_\mathbf{B}} \in C^{\mathcal{I}_\mathbf{B}}$.

$\underline{\alpha \in \mathcal{R}}$:

$\underline{\alpha = s_1 \circ \ldots \circ s_n \sqsubseteq r}$: then for all $a_0, \ldots, a_n \in \Delta^{\mathcal{I}}$, if $(a_0,a_1) \in s_1^{\mathcal{I}} \wedge \ldots \wedge (a_{n-1}, a_n) \in s_n^{\mathcal{I}} \rightarrow (a_0,a_n) \in r^{\mathcal{I}}$. If $(a_i, a_{i+1}) \in s_{i+1}^{\mathcal{I}}$ we find $s_{i+1}(a_i, a_{i+1}) \in \mathbf{B}$, for $0 \leq i < n$, and consequently $(a_i, a_{i+1}) \in s_{i+1}^{\mathcal{I}_\mathbf{B}}$, as well as $(a_0,a_1) \in s_1^{\mathcal{I}}$ yields $r(a_0,a_n) \in \mathbf{B}$ sucht that $(a_0,a_n) \in r^{\mathcal{I}_\mathbf{B}}$.

$\underline{\alpha = \mathsf{Ref}(r)}$: then for all $a \in \Delta^{\mathcal{I}}$ we have $(a,a) \in r^{\mathcal{I}}$, and therefore $r(a,a) \in \mathbf{B}$ such that $(a,a) \in r^{\mathcal{I}_\mathbf{B}}$.

$\underline{\alpha = \mathsf{Irr}(r)}$: then for all $a \in \Delta^{\mathcal{I}}$ we have $(a,a) \notin r^{\mathcal{I}}$, and therefore $\neg r(a,a) \in \mathbf{B}$ such that $(a,a) \notin r^{\mathcal{I}_\mathbf{B}}$.

$\underline{\alpha = \mathsf{Sym}(r)}$: then for all $a, b \in \Delta^{\mathcal{I}}$, if $(a,b) \in r^{\mathcal{I}} \rightarrow (b,a) \in r^{\mathcal{I}}$. If $(a,b) \in r^{\mathcal{I}}$ we have $r(a,b) \in \mathbf{B}$ and consequently $(a,b) \in r^{\mathcal{I}_\mathbf{B}}$. As well as $(b,a) \in r^{\mathcal{I}}$ yields $r(b,a) \in \mathbf{B}$ such that $(b,a) \in r^{\mathcal{I}_\mathbf{B}}$.

Role assertions Asy, Tra and Inv are treated analogously.

$\alpha = (C \sqsubseteq D) \in \mathcal{T}$: then for all $a \in \Delta^{\mathcal{I}}$, if $a \in C^{\mathcal{I}} \rightarrow a \in D^{\mathcal{I}}$, with $C, D \in \mathbf{C}$. As shown, we obtain $a \in C^{\mathcal{I}_\mathbf{B}}$ if $a \in C^{\mathcal{I}}$, for $C \in \mathbf{C}$ and all named individuals a. □

Lemma 3.8. *Let $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$ be a $\mathcal{SROIQ}$ knowledge base. Then for any ground interpretation $\mathcal{I}_\mathbf{B}$ induced by some $\mathbf{B}$ over $\mathcal{K}$, it holds that, if $\mathcal{I}_\mathbf{B} \models \mathcal{K}$ then $\mathcal{I}_\mathbf{B}$ uniquely determines a bounded model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$.*

Proof. $\mathcal{I}_\mathbf{B}$ fulfills the conditions imposed in Definition 3.1. This is:

- $\Delta^{\mathcal{I}_\mathbf{B}} = \{a \in N_I^{dl} \mid a \text{ occurs in } \mathbf{B}\} = \{a \mid a \in N_I^{dl}(\mathcal{K})\} = \Delta^{\mathcal{I}} \neq \emptyset$, as by definition of $\mathbf{B}$, every individual a occurring in $\mathcal{K}$ also occurs in at least one assertion in $\mathbf{B}$.
- For each individual $a \in \Delta^{\mathcal{I}_\mathbf{B}}$, $a^{\mathcal{I}_\mathbf{B}} = a = a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ for any bounded model $\mathcal{I}$.
- For each concept name $A \in N_C^{dl}$, $A^{\mathcal{I}_\mathbf{B}} := \{a \mid A(a) \in \mathbf{B}\} \subseteq \Delta^{\mathcal{I}_\mathbf{B}} = \Delta^{\mathcal{I}}$.
- For each role name $r \in N_R^{dl}$, $r^{\mathcal{I}_\mathbf{B}} := \{(a, b) \mid r(a, b) \in \mathbf{B}\} \subseteq \Delta^{\mathcal{I}_\mathbf{B}} \times \Delta^{\mathcal{I}_\mathbf{B}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.

□

We will only use the notion of *bounded model* in the remainder of this work. As we have just demonstrated, bounded interpretations can be described via some finite set of assertions, which is the representation of choice for our approach described in the next Chapter.

3.3 Reasoning Tasks

As introduced for the standard semantics in Chapter 2.1.5, all standard reasoning tasks carry over to reasoning with respect to bounded models which we will briefly describe. We will then look into more natural questions arising from our ubiquitous motivation of *constraint satisfaction problems* modeled in $\mathcal{SROIQ}$ knowledge bases, and refer to them as *non-standard reasoning taks*.

3.3.1 Standard Reasoning Taks

In the sequel, let $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$ be a $\mathcal{SROIQ}$ knowledge base, $C, D \in \mathbf{C}$ concept descriptions, and $a \in N_I^{dl}$ some named individual.

Satisfiability $\mathcal{K}$ is *satisfiable* w.r.t. the *bounded model semantics*, iff there exists an interpretation $\mathcal{I}$ which is a bounded model of $\mathcal{K}$, $\mathcal{I} \models \mathcal{K}$.

Entailment An axiom α is *entailed* by $\mathcal{K}$ w.r.t. the bounded model semantics, $\mathcal{K} \models \alpha$, iff every bounded model of $\mathcal{K}$ is also a model of α. Reduced to *unsatisfiability*, $\mathcal{K} \models \alpha$ holds, iff $\mathcal{K} \cup \{\neg\alpha\}$ is *unsatisfiable*.

Concept Satisfiability C is *satisfiable* w.r.t. $\mathcal{K}$ under the *bounded model semantics*, iff $C^{\mathcal{I}} \neq \emptyset$ for some bounded model $\mathcal{I}$ of $\mathcal{K}$.

Instance Retrieval For some concept C, *instance retrieval* denotes the task of asking for all individuals a, such that $a^{\mathcal{I}} \in C^{\mathcal{I}}$ holds in every bounded model $\mathcal{I}$.

ABox Consistency An ABox $\mathcal{A}$ is *consistent* w.r.t. $\mathcal{R}$ and $\mathcal{T}$, iff there exists a bounded model $\mathcal{I}$ of $\mathcal{A}$ which is also a model of $\mathcal{R}$ and $\mathcal{T}$.

3.3.2 Non-Standard Reasoning Tasks

We argued that our main motivation is to accommodate the circumstance of dealing with knowledge bases representing a formal problem description as well as a concrete instance of the specified problem. It is therefore quite natural to ask not only whether the knowledge base is satisfiable by some model, but rather an adequate one; i.e. a solution for the encoded problem. Our notion of a *bounded model* already is a first step towards such intuitive models. Consequently, one naturally also is interested in all solutions, which actually reduces to the iterative task of enumerating every (bounded) model of the given knowledge base. In the sequel we describe these two task, which we will refer to as *model extraction* and *model enumeration*.

Model Extraction

Usually, when one is interested in testing satisfiability, the model constructed by the reasoner of choice merely serves as witness to claim satisfiability, rather than an accessible artifact; which, for the purpose of satisfiability testing, is absolutely sufficient. Now, with *model extraction* we refer to the task of materializing an identified model in order to be able to work with it; i.e. the ability to inspect the model in full detail and reuse it in downstream processes. This simply can be expressed by reformulating the satisfiability task in such a way, that instead of answering the satisfiability question with *yes* or *no*, it is answered positively by returning a model and negatively if no model can be returned. With respect to the introduced notions, we can ask for a set of assertions $\mathbf{B} = \mathit{extract}(\mathcal{K})$, and denote $\mathit{extract}(\mathcal{K})$ for some knowledge base $\mathcal{K}$ as follows:

$$\mathit{extract}(\mathcal{K}) := \begin{cases} \mathbf{B}, & \mathcal{I}_{\mathbf{B}} \models \mathcal{K} \text{ for some } \mathbf{B}, \\ \emptyset, & \mathcal{I}_{\mathbf{B}} \not\models \mathcal{K} \text{ for all } \mathbf{B} \end{cases} \tag{3.9}$$

where $\mathbf{B}$ is constructible over $\mathcal{K}$ as shown in Definition 3.6. We do not claim that existing approaches are not able to meet this requirement, quite contrary, we already saw that tableaux algorithms successively construct an ABox, such that a model can be extracted from any open and complete ABox (cf. Section 3.2.1). However, to the best of our knowledge most of the existing tableaux reasoners do not reveal the constructed model, and we therefore merely want to point out the importance of model extraction in the application domain we motivate our approach with.

Model Enumeration

The natural continuation of model extraction, is to make all models explicit by enumerating each model – *model enumeration*. Intuitively we can reduce this task to model

extraction and therefore successively ask for a model which is different from the ones already obtained, in short:

$$enumerate(\mathcal{K}) := \{\mathbf{B} \mid \mathbf{B} \in \mathcal{B}_{\mathcal{K}} \text{ and } \mathcal{I}_{\mathbf{B}} \models \mathcal{K}\}. \tag{3.10}$$

Example 3.9. *Reconsidering* $\mathcal{K}_{3Col} = (\mathcal{R}_{3Col}, \mathcal{T}_{3Col}, \mathcal{A}_{sim})$ *from Example 3.2, one naturally is interested in all possible colorings of the nodes in the given input graph in* $\mathcal{A}_{sim}$*. For the simple graph encoded in* $\mathcal{A}_{sim}$ *there are 12 possible colorings. Consequently we would expect 12 models, all different with respect to the extensions of* RNode, BNode *and* GNode.

3.4 Complexity of Bounded Model Reasoning

Reasoning in $\mathcal{SROIQ}$ over arbitrary interpretations is known to be N2ExpTime-complete [Kaz08]. Though, it is considered to be usable in practice since worst-case knowledge bases would be of very artificial nature. It is therefore not surprising that restricting interpretations to be individual bounded and therefore finite and of known size yields better complexity bounds. We are looking now into the complexity of deciding whether a given knowledge base $\mathcal{K}$ is satisfiable w.r.t. the bounded model semantics, and denote the class of these knowledge bases as:

$$\mathsf{SAT}_{\mathsf{bm}} := \{\mathcal{K} \mid \text{there exists a bounded model } \mathcal{I} \text{ for } \mathcal{K}\}.$$

We already saw an encoding of the graph coloring problem, where even a much less expressive sub-language of $\mathcal{SROIQ}$ was utilized, which we could already use to show NP-hardness. However, for good reasons we use SAT to show the intended results since it contributes to our approach.

Proposition 3.10 ($\mathsf{SAT} \preceq_\rho \mathsf{SAT}_{\mathsf{bm}}$)**.** SAT *is polynomially reducible to* $\mathsf{SAT}_{\mathsf{bm}}$*, i.e. any propositional formula* φ *can be encoded in polynomial time into a* $\mathcal{SROIQ}$ *knowledge base* $\mathcal{K}_\varphi$*, such that* φ *is satisfiable iff* $\mathcal{K}_\varphi$ *is satisfiable.*

Proof. Let φ be a propositional formula in conjunctive normal form; i.e. φ is a conjunction of clauses $C_1 \wedge C_2 \wedge \cdots \wedge C_n$, $n \geq 0$. Each clause is a disjunction of literals $L_1 \vee \cdots \vee L_m$, $m \geq 0$. A corresponding knowledge base $\mathcal{K}_\varphi$ is obtained by directly encoding each clause in φ into an axiom of $\mathcal{K}_\varphi$:

$$\mathcal{K}_\varphi := \left\{ \top(a) \cup \bigcup_{1 \leq i \leq n} \top \sqsubseteq \bigsqcup_{1 \leq j \leq m} \left\{ \begin{array}{ll} Q & \text{, if } L_j \in C_i = q \\ \neg Q & \text{, if } L_j \in C_i = \neg q \end{array} \right\} \right.$$

Beside axioms for each clause in φ, the individual axiom $\top(a)$ is added, nominating a to be the only named individual occurring in $\mathcal{K}_\varphi$. Apparently the translation runs in linear time, however, the assumed presence of φ in conjunctive normal form exhibits polynomially time bounds.

Claim: φ is satisfiable iff $\mathcal{K}_\varphi$ is satisfiable.

($\Rightarrow$) Let φ be a propositional formula in conjunctive normal form, and $\mathcal{I}_\varphi$ an interpretation consisting of all propositional variables of φ which are mapped to true under $\mathcal{I}_\varphi$, such that $\mathcal{I}_\varphi \models \varphi$ holds. $\mathcal{K}_\varphi$ is the set of DL axioms obtained from the translation given above. We construct an interpretation $\mathcal{I}_\mathcal{K} = (\Delta^{\mathcal{I}_\mathcal{K}}, \cdot^{\mathcal{I}_\mathcal{K}})$ for $\mathcal{K}_\varphi$ from $\mathcal{I}_\varphi$ as follows:

- $\Delta^{\mathcal{I}_\mathcal{K}} = \{a\}$, with $a^{\mathcal{I}_\mathcal{K}} = a$, and
- $Q^{\mathcal{I}_\mathcal{K}} = \{a\}$ iff $q \in \mathcal{I}_\varphi$, for every concept name $Q \in N_C^{dl}(\mathcal{K}_\varphi)$ and its corresponding propositional variable q.

We show that $\mathcal{I}_\mathcal{K}$ indeed is a bounded model. Suppose towards contradiction $\mathcal{I}_\mathcal{K} \not\models \mathcal{K}_\varphi$. Then there exists an axiom $\alpha \in \mathcal{K}_\varphi$ and $\mathcal{I}_\mathcal{K} \not\models \alpha$, let $\alpha = Q_1 \sqcup \cdots \sqcup Q_m$, obtained by a clause in φ, say $C = L_1 \vee \cdots \vee L_m$, $m \geq 0$. By construction we have that $Q_i = Q$, if $L_i = q$ and $Q_i = \neg Q$, if $L_i = \neg q$, $1 \leq i \leq m$. Since $\mathcal{I}_\varphi \models \varphi$, there is at least one literal L_x in C which evaluates to true under $\mathcal{I}_\varphi$, $x \in \{1, \ldots, m\}$. Hence,

- if $L_x = \neg q$ then $q \notin \mathcal{I}_\varphi$ and therefore $Q_x = \neg Q$ with $\neg Q^{\mathcal{I}_\mathcal{K}} = \Delta^{\mathcal{I}_\mathcal{K}} \setminus Q^{\mathcal{I}_\mathcal{K}} = \{a\}$ and $\alpha^{\mathcal{I}_\mathcal{K}} \neq \emptyset$ which contradicts the assumption $\mathcal{I}_\mathcal{K} \not\models \alpha$.
- if $L_x = q$ then $q \in \mathcal{I}_\varphi$ and therefore $Q_x = Q$, with $Q^{\mathcal{I}_\mathcal{K}} = \{a\}$ such that $\alpha^{\mathcal{I}_\mathcal{K}} \neq \emptyset$ which contradicts the assumption $\mathcal{I}_\mathcal{K} \not\models \alpha$.

($\Leftarrow$) Let $\mathcal{K}_\varphi$ be obtained from some propositional formula φ in conjunctive normal form, as shown. Let $\mathcal{I}_\mathcal{K} = (\Delta^{\mathcal{I}_\mathcal{K}} = \{a\}, \cdot^{\mathcal{I}_\mathcal{K}})$ be a bounded model for $\mathcal{K}_\varphi$, $\mathcal{I}_\mathcal{K} \models \mathcal{K}_\varphi$. Let φ be the propositional formula which $\mathcal{K}_\varphi$ originated from. We construct an interpretation $\mathcal{I}_\varphi$ from $\mathcal{I}_\mathcal{K}$ as follows

$$\mathcal{I}_\varphi := \{q \mid \text{if } Q^{\mathcal{I}_\mathcal{K}} \neq \emptyset, \text{ for each } Q \in N_C^{dl}(\mathcal{K}_\varphi)\},$$

and show that $\mathcal{I}_\varphi$ is indeed a model for φ. Towards contradiction suppose $\mathcal{I}_\varphi \not\models \varphi$. Then there exists an unsatisfied clause $C = L_1 \vee \cdots \vee L_m$. Let $\alpha \in \mathcal{K}_\varphi$ be the axiom obtained from C. Since $\mathcal{I}_\mathcal{K} \models \alpha$; i.e. $\alpha^{\mathcal{I}_\mathcal{K}} \neq \emptyset$, there must be a concept description $Q_x \in \alpha$, $x \in \{1, \ldots, m\}$, such that ${Q_x}^{\mathcal{I}_\mathcal{K}} \neq \emptyset$. Hence,

- if $Q_x = Q$ then $Q^{\mathcal{I}_\mathcal{K}} \neq \emptyset$ and by construction $q \in \mathcal{I}_\varphi$ which satisfies C. Contradiction.
- if $Q_x = \neg Q$ then $\neg Q^{\mathcal{I}_\mathcal{K}} = \Delta^{\mathcal{I}_\mathcal{K}} \setminus Q^{\mathcal{I}_\mathcal{K}} \neq \emptyset$ and by construction $q \notin \mathcal{I}_\varphi$ which satisfies C. Contradiction.

□

Example 3.11. *To illustrates the proposed translation, let* $\varphi = (\neg p) \wedge (q \vee r) \wedge (\neg r \vee p) \wedge s$*, which is obviously satisfiable, and there is exactly one interpretation evaluating* φ *to be true,* $\mathcal{I} = \{s, q\}$*. Accordingly we obtain*

$$\mathcal{K}_\varphi = \{\top(a),\ \top \sqsubseteq \neg R \sqcup P,\ \top \sqsubseteq Q \sqcup R,\ \top \sqsubseteq \neg P, \top \sqsubseteq S\}.$$

It is not hard to see, that in fact there is only one bounded model, satisfying $\mathcal{K}_\varphi$, which is $\mathcal{I} = \{\Delta^\mathcal{I}, \cdot^\mathcal{I})$, with $\Delta^\mathcal{I} = \{a\}$, is the individual bounded domain, s.t. $a^\mathcal{I} = a$, and $S^\mathcal{I} = Q^\mathcal{I} = \{a\}$.

We have shown correspondence of models of a propositional formula and its DL translation. Since the obtained DL axioms do by far not use expressive extensions of $\mathcal{SROIQ}$, it is at least not obvious that we can establish the same result in the other direction; i.e. translate any $\mathcal{SROIQ}$ knowledge base into a propositional formula, preserving model correspondence. In the sequel we will provide such a translation in order to show the reducibility of SAT_bm to SAT, $\mathsf{SAT}_\mathsf{bm} \preceq_p \mathsf{SAT}$.

Proposition 3.12 ($\mathsf{SAT}_\mathsf{bm} \preceq_p \mathsf{SAT}$)**.** *Any $\mathcal{SROIQ}$ knowledge base $\mathcal{K}$ can be encoded in polynomial time to a propositional formula $\varphi_\mathcal{K}$, $\mathsf{SAT}_\mathsf{bm} \preceq_p \mathsf{SAT}$, such that $\mathcal{K}$ is satisfiable w.r.t. some bounded model iff $\varphi_\mathcal{K}$ is satisfiable.*

Proof. As introduced in Chapter 2.1.3 there exists a direct first-order translation $\tau(\mathcal{K})$ for any $\mathcal{SROIQ}$ knowledge base $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$. We can denote the bounded domain induced by $\mathcal{K}$ also as finite Herbrand universe $\mathcal{U}_{\tau(\mathcal{K})}$ of $\tau(\mathcal{K})$, since individuals correspond to constants in $\tau(\mathcal{K})$ and are the only function symbols occurring. Hence, the notion of bounded model obviously carries over to the first-order case. Moreover, $\mathcal{B}_{\tau(\mathcal{K})}$ is the Herbrand base that can be built of predicate names in $\tau(\mathcal{K})$ with the constants in $\mathcal{U}_{\tau(\mathcal{K})}$ as arguments. It is possible to directly ground instantiate $\mathcal{K}$ by adapting the first-order translation, as depicted in Table 3.1. To ensure grounding in polynomial time, we need to tweak role chain axioms, and do so by stripping them down to less complex ones of fixed length. In consequence the propositional equivalent $\tau_{gnd}(\mathcal{K})$ is obtained. Then, for any subset $\mathcal{I}_B \subseteq \mathcal{B}_{\tau(\mathcal{K})}$ we have that $\mathcal{I}_B$ is a Hebrand model for $\tau_{gnd}(\mathcal{K})$ if and only if $\mathcal{I}'_B = (\mathcal{U}_{\tau(\mathcal{K})}, \cdot^{\mathcal{I}'_B})$ is a bounded model for $\tau(\mathcal{K})$, where $\cdot^{\mathcal{I}'_B}$ is obtained from $\mathcal{I}_B$ as shown in Lemma 3.8 and Lemma 3.7 for the other direction. □

Theorem 3.13 (SAT_bm is NP-complete)**.**

Proof. We identify NP-membership in Proposition 3.12 and obtain NP-hardness from Proposition 3.10. □

As observed in Proposition 3.10, we can translate any propositional formula into DL axioms. However, the obtained axioms do not use the full expressivity of $\mathcal{SROIQ}$ at all, they merely correspond to the DL $\mathcal{ALC}$, and do not even use roles. We can identify, that $\mathcal{SROIQ}$ looses its actual expressive means and obeys an expressivity on the level of $\mathcal{ALC}$. We are not discussing this finding further at this point, but will take it up again in Section 6, as it exhibits an interesting use case.

$\tau_{gnd}(\mathcal{K})$	$\bigwedge_{\alpha\in\mathcal{K}} \tau_{gnd}(\alpha)$
Axioms	
$\tau_{gnd}(C \sqsubseteq D)$	$\bigwedge_{\delta\in\Delta}(\tau_{gnd}(C,\delta) \rightarrow \tau_{gnd}(D,\delta))$
$\tau_{gnd}(A(a))$	$A(a)$
$\tau_{gnd}(r(a,b) \mid a = b \mid a \neq b)$	$r(a,b) \mid a = b \mid a \neq b$
$\tau_{gnd}(s_1 \circ \cdots \circ s_n \sqsubseteq r)$	$\bigwedge_{i=1}^{n-1} \tau(s_i \circ s_{i+1} \sqsubseteq s_{i,i+1})$
$\tau_{gnd}(s_1 \circ \cdots \circ s_n \sqsubseteq r)$	$\bigwedge_{\{\delta,\delta_1,\delta_2\}\in\Delta} s_1(\delta,\delta_1) \wedge s_2(\delta_1,\delta_1) \rightarrow r(\delta,\delta_2)$
$\tau_{gnd}(\mathsf{Irr}(r) \mid \mathsf{Ref}(r))$	$\bigwedge_{\delta\in\Delta} \neg r(\delta,\delta) \mid \bigwedge_{\delta\in\Delta} r(\delta,\delta)$
$\tau_{gnd}(\mathsf{Tra}(r))$	$\bigwedge_{\delta_1,\delta_2,\delta_3\in\Delta} r(\delta_1,\delta_2) \wedge r(\delta_2,\delta_3) \rightarrow r(\delta_1,\delta_3)$
$\tau_{gnd}(\mathsf{Sym}(r))$	$\bigwedge_{\delta_1,\delta_2\in\Delta} r(\delta_1,\delta_2) \rightarrow r(\delta_2,\delta_1)$
$\tau_{gnd}(\mathsf{Dis}(r,s))$	$\bigwedge_{\delta_1,\delta_2\in\Delta} r(\delta_1,\delta_2) \wedge s(\delta_1,\delta_2) \rightarrow \bot$
$\tau_{gnd}(\mathsf{Asy}(r))$	$\bigwedge_{\delta_1,\delta_2\in\Delta} r(\delta_1,\delta_2) \wedge r(\delta_2,\delta_1) \rightarrow \bot$
Concept Expressions	
$\tau_{gnd}(A,t)$	$A(t)$
$\tau_{gnd}(\neg C,t)$	$\neg\tau(C,t)$
$\tau_{gnd}(C \sqcup D,t)$	$\tau_{gnd}(C,t) \vee \tau_{gnd}(D,t)$
$\tau_{gnd}(C \sqcap D,t)$	$\tau_{gnd}(C,t) \wedge \tau_{gnd}(D,t)$
$\tau_{gnd}(\forall r.C,t)$	$\bigwedge_{\delta\in\Delta}(\neg r(t,\delta) \vee \tau_{gnd}(C,\delta))$
$\tau_{gnd}(\exists r.C,t)$	$\bigvee_{\delta\in\Delta}(r(t,\delta) \wedge \tau_{gnd}(C,\delta))$
$\tau_{gnd}(\exists r.Self,t)$	$r(t,t)$
$\tau_{gnd}(\{a\},t)$	$O_a(a) \wedge a = t$
$\tau_{gnd}(\leq n\ r.C,t)$	naive at-most sat encoding.
$\tau_{gnd}(\geq n\ r.C,t)$	naive at-least sat encoding.

Table 3.1: Grounded First-Order Translation (cf. 2.1.3).

3.5 Tableaux based Bounded Model Reasoning

We want explore now how *tableaux based reasoners* perform on *bounded model reasoning*. It is possible to utilize axiom $\top \equiv \{a_1,\ldots,a_n\}$ to enforce a singleton domain as well as binding the extension of each concept to any subset of $\{a_1,\ldots,a_n\}$. In fact, such a restriction on the $\top$-concept can be used in any knowledge base in order to employ existing reasoning services to perform reasoning on bounded domains.

In detail, consider the axiom $\top \equiv \{a_1,\ldots,a_n\}$, with $a_i \in N_I^{dl}$, $1 \leq i \leq n$, which can be represented by $\top \sqsubseteq \{a_1,\ldots,a_n\}$ (1) and $\{a_1,\ldots,a_n\} \sqsubseteq \top$ (2). Where (1) is often referred to as *oneOf*-axiom, such an enumeration of individuals realizes a restriction on the subsuming concept (here $\top$), such that for any $a^{\mathcal{I}} \in \top^{\mathcal{I}}$ implies $a^{\mathcal{I}} = a_i^{\mathcal{I}}$ for exactly one $i \in \{1,\ldots,n\}$. Consequently, the nominal concept $\{a_1,\ldots,a_n\}$ is equivalent

to the concept expression $\{a_1\} \sqcup \cdots \sqcup \{a_n\}$. Axiom (2) imposes the restriction in the other direction, and by definition we have $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$, as well as $\top^{\mathcal{I}} \subseteq \{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\}$ and $\{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\} \subseteq \top^{\mathcal{I}}$ must hold for any model $\mathcal{I}$, which demonstrates the enforced domain bound quite well. It is easy to see, and stated without proof, that for any knowledge base $\mathcal{K}$ and (standard) interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, it holds:

$$\mathcal{I} \models \mathcal{K} \cup \{\top \equiv \{a_1, \ldots, a_n\}\} \iff \mathcal{I} \text{ is a bounded model for } \mathcal{K},$$

with $\{a_1, \ldots, a_n\} = N_I^{dl}(\mathcal{K})$.[3] Thus, adding the $\top$-concept restriction, any reasoner is performing reasoning w.r.t. bounded models.

Apparently individual enumerations, especially in our case, exhibit a potentially large disjunction. For tableaux based reasoners, restricting the $\top$-concept in this way leads to a significant number of additional backtracking points. As shown in Sirin et al. [Sir06], it is possible to use absorption to get rid of axiom (2). It uses an equivalent transformation, where $\{a\} \sqsubseteq C$ is equivalent to the assertion $C(a)$. Applied to $\top \sqsubseteq \{a_1, \ldots, a_n\}$, we are left with GCI $\top \sqsubseteq \{a_1, \ldots, a_n\}$ and n assertions $\top(a_1); \ldots; \top(a_n)$. However, since we deal with the $\top$-concept, the introduced assertions are not necessary as they implicitly hold by definition of the semantics anyway, and we therefore can focus on the axiom $\top \sqsubseteq \{a_1, \ldots, a_n\}$ only – which enforces in fact the same behavior as $\top \equiv \{a_1, \ldots, a_n\}$. For some named concept A, an *oneOf*-axiom $A \sqsubseteq \{a_1, \ldots, a_n\}$ would not be that critical, since its a local restriction and affects only instances of A, whereas in case of the $\top$-concept any instance of any concept is embraced and has global influence.

Example 3.14. *We intend to construct an unsatisfiable knowledge base $\mathcal{K}_n = (\emptyset, \mathcal{T}_n, \mathcal{A}_n)$, with $\mathcal{T}_n$ and $\mathcal{A}_n$ specified as follows:*

$$\begin{aligned} \mathcal{T}_n &= \{A_1 \sqsubseteq \exists r.A_2, \ldots, A_n \sqsubseteq \exists r.A_{n+1}, A_{n+1} \sqsubseteq \top\} \cup \\ &\quad \{A_i \sqcap A_j \sqsubseteq \bot \mid 1 \leq i < j \leq n+1\} \cup \\ &\quad \{\top \sqsubseteq \{a_1, \ldots, a_n\}\} \\ \mathcal{A}_n &= \{A_1(a_1)\} \end{aligned}$$

The idea is to encode a finite chain of instances of concepts A_i, $1 \leq i \leq n$, where every instance of A_i is required to have some r-successor in A_{i+1}, except for instances of A_{n+1} representing the chain end. By asserting a_1 to A_1 we enforce a non-empty chain starting at a_1. Intentionally, we bind the domain to exactly n individuals – one less than satisfiability in this case requires. Figure 3.5 depicts the input scenario.

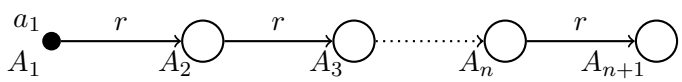


Figure 3.1: Unsatisfiable Chain-Scenario.

[3] This is up to renaming, such that the condition of mapping each individual to itself is met.

In order to claim unsatisfiability of $\mathcal{K}_n$, the reasoner is now doomed to test for any individual $a_i \in \{a_1, \ldots, a_n\}$ its membership in $A_2, \ldots, A_{n+1}$, which will eventually fail due to the disjoint axioms. Without illustrating the full execution of the tableaux algorithm on $\mathcal{K}_n$, first assume optimistically that memberships $A_2(a_2), \ldots, A_n(a_n)$ are explored in $n-1$ steps.[4] Then applying the $\exists$-rule for $A_n \sqsubseteq \exists r.A_{n+1}$ with $A_n(a_n)$ eventually fails n-times in the attempt of introducing $r(a_n, a_i)$ and $A_{n+1}(a_i)$, $1 \leq i \leq n$, which requires to backtrack until the decision of introducing $r(a_{n-1}, a_n)$ and $A_n(a_n)$. This leads again to $n-1$ attempts of introducing $r(a_{n-1}, a_i)$ and $A_n(a_i)$, $1 \leq i \leq n-1$. Essentially backtracking repeats failing ultimately for each of the $n-1$ remaining decisions of introducing $r(a_1, a_i)$ and $A_2(a_i)$, $1 \leq i \leq n$ and $i \neq 2$. Finally, unsatisfiability can be detected after all n^n possibilities are tested.

The type of problems we are considering in this work are exactly these combinatorial problems. Where nominals itself are not the issue, rather the resulting disjunctions when using nominals in the mentioned way. Any DL reasoner naturally tries to avoid OR-branching whenever possible; so does HermiT [MSH09] by sophisticated pre-processing of the knowledge base into DL-clauses in order to explore determinism where an allegedly non-deterministic choice would have been made by naive tableaux rule application. However, to the best of our knowledge, none of the known reasoning algorithms is designed with a strong focus on combinatorial aspects, which leads to the conclusion that employing existing reasoners to perform bounded model reasoning is not practical.[5]

3.5.1 Model Extraction and Enumeration

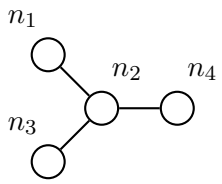
With *model extraction* we emphasized the need of having a computed model available as usable object. Tableaux algorithms successively try to construct models. From a theoretical perspective it seems therefore an easy and straight-forward task just to read off the model from any complete and open ABox constructed by the tableaux, but it turns out to be not possible for $\mathcal{SROIQ}$ reasoners without having access to the actual implementation and its internal data structures. In consequence we can also not hope for an enumeration of all models. In the next chapter we will propose a quite different computational approach to compute bounded models.

[4] Optimistic in the sense that we assume already an individual reuse strategy, that is that an application of the $\exists$-rule is reusing individuals already occurring in the current branch, if possible, before introducing new ones.

[5] We will underpin these statements with empirical results in Chapter 5.

3.6 Discussion

Are bounded models reasonable models? Lets reconsider the graph coloring problem from Example 3.2, and an alternative input graph $\mathcal{A}_{sim'}$, of the following form:



$$\mathcal{A}_{sim'} = \{ \mathsf{Node}(n_1; n_2; n_3; n_4), \mathsf{edge}(n_1, n_2), \mathsf{edge}(n_2, n_3), \mathsf{edge}(n_2, n_4) \}$$

We argued that any bounded model exhibits a solution for the underlying combinatorial problem. Inspecting every bound model for $\mathcal{K}_{3Col} = (\mathcal{R}_{3Col}, \mathcal{T}_{3Col}, \mathcal{A}_{sim'})$, this is apparently true. However, there are models which one would intuitively not expect, for example, the interpretation $\mathcal{I}_{\mathbf{B}_4} = (\{n_1, \ldots, n_4\}, \cdot^{\mathcal{I}_{\mathbf{B}_4}})$ induced by the following set $\mathbf{B}_4$ is a bounded model.

$$\begin{aligned}\mathbf{B}_4 = \{ & \mathsf{Node}(n_1; n_2; n_3; n_4), \mathsf{edge}(n_1, n_2), \mathsf{edge}(n_2, n_3), \mathsf{edge}(n_2, n_4), \mathsf{edge}(n_1, n_3), \\ & \neg\mathsf{edge}(n_1, n_4), \neg\mathsf{edge}(n_3, n_4), \mathsf{RNode}(n_1; n_4), \neg\mathsf{RNode}(n_2; n_3), \mathsf{BNode}(n_3), \\ & \neg\mathsf{BNode}(n_1; n_2; n_4), \mathsf{GNode}(n_2), \neg\mathsf{GNode}(n_1; n_3; n_4) \}\end{aligned}$$

But it also adds the additional edge (n_1, n_3) to the simple input graph given in $\mathcal{A}_{sim'}$ which certainly does not reflect the intended behavior. One could argue that this is enough misbehavior to refuse the idea of bounded models adequately representing solutions. Contrary, claiming that the given problem instance (here the given graph $\mathcal{A}_{sim'}$) is simply not sufficiently specified is also a valid argument and the position we take up. The graph in $\mathcal{A}_{sim'}$ can be fixed by adding negative assertions $\neg\mathsf{edge}(n_1, n_3)$, $\neg\mathsf{edge}(n_1, n_4)$ and $\neg\mathsf{edge}(n_3, n4)$, excluding models such as $\mathcal{I}_{\mathbf{B}_4}$.

This issue also relates to the question whether an *open world semantics* should be used in the context of such problems. As this example demonstrates, a *closed world* approach where we could conclude $\neg\mathsf{edge}(n_1, n_3)$ from the non-existence of $\mathsf{edge}(n_1, n_3)$, would overcome the requirement of providing the problem instance as *complete world*.

4 Encoding $\mathcal{SROIQ}$ Knowledge Bases

In this chapter we develop an encoding of an arbitrary $\mathcal{SROIQ}$ knowledge base $\mathcal{K}$, into an *answer set program* $\Pi^{\mathcal{K}}$, such that the set of answer sets coincides with the set of bounded models of the given knowledge base. This allows us to perform not only standard reasoning tasks on $\mathcal{K}$, but also the imposed tasks of *model extraction* and *model enumeration* quite elegantly.

4.1 General Idea

Using existing OWL 2 reasoners for bounded model reasoning has been shown in Section 3.5 to be an *unsatisfactory* setup. Not only is satisfiability and unsatisfiability testing infeasible using the suggested workaround, also the task of model extraction and enumeration were shown to be not possible without intensive changes on the reasoning algorithms and their implementations. Instead doing so, we try to reuse existing computational approaches to our purpose of reasoning with respect to bounded models.

Any answer set is a *finite set of ground atoms* representing a model of the logic program under the answer set semantics (cf. Section 2.2.2), as well as bounded models can be represented as *finite set of assertions* (ground atoms) as depicted in Section 3.2.2. This bears the idea of teaming up these two circumstances, and turn the satisfiability problem into a constraint satisfaction problem. We aim for a translation of each axiom in such way, that the resulting logic program has all bounded models as its solutions, and vice versa; and therefore obtain a *decision procedure for bounded model reasoning.*

4.1.1 Candidate Generation

For any knowledge base $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$, we can denote any bounded model of $\mathcal{K}$ via a set of assertions $\mathbf{B}$ constructible over $\mathcal{K}$. Apparently one can construct all sets $\mathbf{B}$, each of them potentially inducing a bounded model $\mathcal{I}_{\mathbf{B}}$. This gives rise to the idea to encode the construction into rules of the logic program such that each set is computed and checked whether it yields a bounded model for $\mathcal{K}$, which corresponds to the guessing part of the guess and check paradigm mentioned in Section 2.2.3.

4.1.2 Candidate Exclusion

Given a set $\mathbf{B}$ of assertions constructible over $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$, one can check whether the interpretation $\mathcal{I}_{\mathbf{B}}$ induced by $\mathbf{B}$ is a bound model of $\mathcal{K}$. Therefore, every axiom of $\mathcal{K}$ is taken as a constraint, such that some $\mathbf{B}$ is excluded in case of constraint violation.

4.2 Normalization

In order to translate axioms adequately into syntactically correct rules it is necessary to break down complex concept expressions. The nested expression $\exists r.(\leq\ n\, s.(A \sqcup B)$, for example, can not be directly translated into a rule. Pre-processing mainly involves *negation normal form* (nnf) passing negation directly on to named concepts, and a variation of the *structural transformation* originally proposed by Plaisted and Greenbaum [PG86], and with respect to GCIs and complex concepts, has been further adapted in [MSH09]. For the previous axiom, the transformation yields $\exists r.A_1$, $A_1 \sqsubseteq \leq n\, s.A_2$, and $A_2 \sqsubseteq A \sqcup B$.

$$\Omega(\mathcal{K}) = \bigcup_{\alpha \in \mathcal{R} \cup \mathcal{A}} \Omega(\alpha) \cup \bigcup_{C_1 \sqsubseteq C_2 \in \mathcal{T}} \Omega(\top \sqsubseteq \mathsf{nnf}(\neg C_1 \sqcup C_2)) \tag{4.1}$$

$$\Omega(\top \sqsubseteq \mathbf{C} \sqcup C') = \Omega(\top \sqsubseteq \mathbf{C} \sqcup \alpha_{C'}) \cup \bigcup_{1 \leq i \leq n} \Omega(\top \sqsubseteq \dot{\neg}\alpha_{C'} \sqcup C_i) \tag{4.2}$$

for C' of the form $C' = C_1 \sqcap \cdots \sqcap C_n$ and $n \geq 2$

$$\Omega(\top \sqsubseteq \mathbf{C} \sqcup \forall r.D) = \Omega(\top \sqsubseteq \mathbf{C} \sqcup \forall r.\alpha_D) \cup \Omega(\top \sqsubseteq \dot{\neg}\alpha_D \sqcup D) \tag{4.3}$$

$$\Omega(\top \sqsubseteq \mathbf{C} \sqcup \geq n\, r.D) = \Omega(\top \sqsubseteq \mathbf{C} \sqcup \geq n\, r.\alpha_D) \cup \Omega(\top \sqsubseteq \dot{\neg}\alpha_D \sqcup D) \tag{4.4}$$

$$\Omega(\top \sqsubseteq \mathbf{C} \sqcup \leq n\, r.D) = \Omega(\top \sqsubseteq \mathbf{C} \sqcup \leq n\, r.\dot{\neg}\alpha_{\dot{\neg}D}) \cup \Omega(\top \sqsubseteq \dot{\neg}\alpha_{\dot{\neg}D} \sqcup \dot{\neg}D) \tag{4.5}$$

$$\Omega(\top \sqsubseteq \mathbf{C} \sqcup \neg\{s\}) = \begin{cases} \bot & \text{if } \mathbf{C} = \emptyset, \\ \Omega(\mathbf{C}(s)) & \text{otherwise.} \end{cases} \tag{4.6}$$

$$\Omega(D(s)) = \{\alpha_D(s)\} \cup \Omega(\top \sqsubseteq \dot{\neg}\alpha_D \sqcup \mathsf{nnf}(D)) \tag{4.7}$$

$$\Omega(r^-(s,t)) = \{r(t,s)\} \tag{4.8}$$

$$\Omega(\beta) = \{\beta\} \text{ for any other axiom } \beta \tag{4.9}$$

$$\alpha_C = \begin{cases} Q_C & \text{if } \mathsf{pos}(C) = \mathsf{true} \\ \neg Q_C & \text{if } \mathsf{pos}(C) = \mathsf{false} \end{cases}, \text{where } Q_C \text{ is a fresh concept name unique for } C$$

$$\begin{aligned}
\mathsf{pos}(\top) &= \mathsf{false} & \mathsf{pos}(\bot) &= \mathsf{false} \\
\mathsf{pos}(A) &= \mathsf{true} & \mathsf{pos}(\neg A) &= \mathsf{false} \\
\mathsf{pos}(\{s\}) &= \mathsf{true} & \mathsf{pos}(\neg\{s\}) &= \mathsf{false} \\
\mathsf{pos}(\exists r.Self) &= \mathsf{true} & \mathsf{pos}(\neg\exists r.Self) &= \mathsf{false} \\
\mathsf{pos}(C_1 \sqcap C_2) &= \mathsf{pos}(C_1) \vee \mathsf{pos}(C_2) & \mathsf{pos}(C_1 \sqcup C_2) &= \mathsf{pos}(C_1) \vee \mathsf{pos}(C_2) \\
\mathsf{pos}(\forall r.C_1) &= \mathsf{pos}(C_1) & \mathsf{pos}(\leq n\, r.C_1) &= \begin{cases} \mathsf{pos}(\dot{\neg}C_1) & \text{if } n = 0 \\ \mathsf{true} & \text{otherwise} \end{cases} \\
\mathsf{pos}(\geq n\, r.C_1) &= \mathsf{true}
\end{aligned} \tag{4.10}$$

Note: A is an atomic concept, $C_{(i)}$ are arbitrary concept expressions, $\mathbf{C}$ is a possibly empty disjunction of concept expressions, D is **not** a literal concept. The function $\dot{\neg}$ is defined as $\dot{\neg}(\neg A) = A$ and $\dot{\neg}(A) = \neg A$ for some atomic concept A.

Table 4.1: Ω-Normalization of knowledge base axioms.

Definition 4.1 (Normalized Form [MSH09]). *A GCI is normalized if it is of the form* $\top \sqsubseteq \bigsqcup_{i=1}^{n} C_i$, *where* C_i *is of the form* B, $\{a\}$, $\forall r.B$, $\exists r.Self$, $\neg\exists r.Self$, $\geq n\, r.B$, *or* $\leq n\, r.B$, *for* B *a literal concept,* r *a role, and* n *a positive integer. A TBox* $\mathcal{T}$ *is normalized, if each GCI in* $\mathcal{T}$ *is normalized. An ABox* $\mathcal{A}$ *is normalized if each concept assertion in* $\mathcal{A}$ *contains only a literal concept, each role assertion in* $\mathcal{A}$ *contains only an atomic role, and* $\mathcal{A}$ *contains at least one assertion. A* $\mathcal{SROIQ}$ *knowledge base* $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$ *is normalized if* $\mathcal{T}$ *and* $\mathcal{A}$ *are normalized.*

Given $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$, the normalized form $\Omega(\mathcal{K})$ is obtained by applying the transformation Ω, given in Table 4.1. Essentially, this is the transformation used in the Hypertableaux algorithm introduced by Motik et al. [MSH09], except that we neglect to introduce any new individuals. Moreover, potential complex concept assertions are rewritten into atomic assertions of the form $A(a)$ and $r(a, b)$. Note that we neglect inequality statements $a \not\approx b$, since by definition of the bounded interpretation, we implicitly adhere to an *unique name assumption*.

The normalized knowledge base $\Omega(\mathcal{K})$ and $\mathcal{K}$ are equisatisfiable; i.e. $\mathcal{K}$ is satisfiable, if and only if $\Omega(\mathcal{K})$ is satisfiable, which is proven in [MSH09], Lemma 2, and carries over to the bounded model semantics. More important, Ω is *structure-preserving* and, although we do not obtain semantic equivalence, it is possible to extract a model for $\mathcal{K}$, given a model of $\Omega(\mathcal{K})$. In the remainder of this chapter, we will assume that some knowledge base $\mathcal{K}$ is in normalized form, if not explicitly stated differently.

Example 4.2 (Graduate Study Problem). *From Example 2.6, consider the following (shortened) axiom:* $\exists \mathsf{completed}.\{\mathsf{krr}\} \sqsubseteq \geq 2\ \mathsf{passed}.(\mathsf{4HCourse} \sqcap (\exists \mathsf{offers}^{-}.\{\mathsf{krr}\}))$, *for which we apply negation normal form transformation (*nnf*) on the first call of* Ω, *and obtain:*

$$\Omega(\top \sqsubseteq (\underline{\forall \mathsf{completed}.\neg\{\mathsf{krr}\}}_{1}) \sqcup (\underline{\geq 2\ \mathsf{passed}.(\mathsf{4HCourse} \sqcap (\exists \mathsf{offers}^{-}.\{\mathsf{krr}\}))}_{2}))$$

For each subscripted sub-expression, Ω *is recursively applied, where the result of a call on sub-expression* i, *is indicated via* $=_i$*:*

$$=_1 \quad \Omega(\top \sqsubseteq \forall \mathsf{completed}.\neg \mathsf{Q}_1 \sqcup (\underline{\ldots}_2)) \cup \Omega(\top \sqsubseteq \mathsf{Q}_1 \sqcup \underline{\neg\{\mathsf{krr}\}}_3) \tag{4.11}$$

$$=_3 \quad \Omega(\top \sqsubseteq \forall \mathsf{completed}.\neg \mathsf{Q}_1 \sqcup (\underline{\ldots}_2)) \cup \Omega(\underline{\mathsf{Q}_1(\mathsf{krr})}_4) \tag{4.12}$$

$$=_4 \quad \Omega(\top \sqsubseteq \forall \mathsf{completed}.\neg \mathsf{Q}_1 \sqcup (\underline{\ldots}_2)) \cup \mathsf{Q}_1(\mathsf{krr}) \tag{4.13}$$

$$=_2 \quad \Omega(\top \sqsubseteq \forall \mathsf{completed}.\neg \mathsf{Q}_1 \sqcup (\geq 2\ \mathsf{passed}.\mathsf{Q}_2)) \cup \tag{4.14}$$
$$\Omega(\top \sqsubseteq \neg \mathsf{Q}_2 \sqcup (\underline{\mathsf{4HCourse} \sqcap (\exists \mathsf{offers}^{-}.\{\mathsf{krr}\})}_5)) \cup \mathsf{Q}_1(\mathsf{krr})$$

$$=_5 \quad \Omega(\top \sqsubseteq \forall \mathsf{completed}.\neg \mathsf{Q}_1 \sqcup (\geq 2\ \mathsf{passed}.\mathsf{Q}_2)) \cup \tag{4.15}$$
$$\Omega(\top \sqsubseteq \neg \mathsf{Q}_2 \sqcup \mathsf{Q}_3) \cup \Omega(\top \sqsubseteq \neg \mathsf{Q}_3 \sqcup \mathsf{4HCourse}) \cup$$
$$\Omega(\top \sqsubseteq \neg \mathsf{Q}_3 \sqcup (\underline{\exists \mathsf{offers}^{-}.\{\mathsf{krr}\}}_6)) \cup \mathsf{Q}_1(\mathsf{krr})$$

$$=_6 \quad \Omega(\top \sqsubseteq \forall \mathsf{completed}.\neg \mathsf{Q}_1 \sqcup (\geq 2\ \mathsf{passed}.\mathsf{Q}_2)) \cup \tag{4.16}$$
$$\Omega(\top \sqsubseteq \neg \mathsf{Q}_2 \sqcup \mathsf{Q}_3) \cup \Omega(\top \sqsubseteq \neg \mathsf{Q}_3 \sqcup \mathsf{4HCourse}) \cup$$
$$\Omega(\top \sqsubseteq \neg \mathsf{Q}_3 \sqcup (\leq 1\ \mathsf{offers}^{-}.\mathsf{Q}_4)) \cup \Omega(\top \sqsubseteq \mathsf{Q}_4 \sqcup \underline{\neg\{\mathsf{krr}\}}_7) \cup \mathsf{Q}_1(\mathsf{krr})$$

$$=_7 \quad \Omega(\top \sqsubseteq \forall \mathsf{completed}.\neg \mathsf{Q}_1 \sqcup (\geq 2\ \mathsf{passed}.\mathsf{Q}_2)) \cup \tag{4.17}$$
$$\Omega(\top \sqsubseteq \neg \mathsf{Q}_2 \sqcup \mathsf{Q}_3) \cup \Omega(\top \sqsubseteq \neg \mathsf{Q}_3 \sqcup \mathsf{4HCourse}) \cup$$

$$\begin{aligned} & \Omega(\top \sqsubseteq \neg \mathsf{Q}_3 \sqcup (\leq 1\ \mathsf{offers}^{-}.\mathsf{Q}_4)) \cup \Omega(\mathsf{Q}_4(\mathsf{krr})) \cup \mathsf{Q}_1(\mathsf{krr}) \\ = \quad & \{\top \sqsubseteq \forall \mathsf{completed}.\neg \mathsf{Q}_1 \ \sqcup \ (\geq 2\ \mathsf{passed}.\mathsf{Q}_2),\ \top \sqsubseteq \neg \mathsf{Q}_2 \sqcup \mathsf{Q}_3, \\ & \top \sqsubseteq \neg \mathsf{Q}_3 \sqcup \mathsf{4HCourse},\ \top \sqsubseteq \neg \mathsf{Q}_3 \sqcup (\leq 1\ \mathsf{offers}^{-}.\mathsf{Q}_4),\ \mathsf{Q}_4(\mathsf{krr}),\ \mathsf{Q}_1(\mathsf{krr})\} \end{aligned} \tag{4.18}$$

Concept names $\mathsf{Q}_1, \ldots, \mathsf{Q}_4$ *were introduced and added to the vocabulary.*

4.3 Naïve Encoding

With a normalized knowledge base $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$ at hand, we will now develop a modular encoding into an *answer set program*. We stick to the general *guess and check paradigm* and therefore propose a sub-program $\Pi_{gen}(\mathcal{K})$ generating bounded interpretations as well as turning TBox and RBox axioms into a constraining sub-program $\Pi_{chk}(\mathcal{K})$, ruling out potential candidate interpretations and thereby forming the set of bounded models. As a result, we obtain the program $\Pi(\mathcal{K}) = \Pi_{gen}(\mathcal{K}) \cup \Pi_{chk}(\mathcal{K})$.

4.3.1 Candidate Generation

As shown, any potential bounded interpretation $\mathcal{I}_{\mathbf{B}}$ is induced by a set of individual assertions $\mathbf{B}$, such that for each concept name A, role name r and individuals a, b occurring in $\mathcal{K}$, either $A(a) \in \mathbf{B}$, or $\neg A(a) \in \mathbf{B}$ and either $r(a,b) \in \mathbf{B}$ or $\neg r(a,b) \in \mathbf{B}$. This construction is straight forward to encode via subsequent rules:

$$\begin{aligned} \Pi_{gen}(\mathcal{K}) \quad := \quad & \{\mathsf{A}(X) :\!-\ \mathsf{not}\ \neg \mathsf{A}(X),\ \top(X) \mid A \in N_C^{dl}(\mathcal{K})\} \cup && (4.19) \\ & \{\neg \mathsf{A}(X) :\!-\ \mathsf{not}\ \mathsf{A}(X),\ \top(X) \mid A \in N_C^{dl}(\mathcal{K})\} \cup && (4.20) \\ & \{\mathsf{r}(X,Y) :\!-\ \mathsf{not}\ \neg \mathsf{r}(X,Y), \top(X), \top(Y) \mid r \in N_R^{dl}(\mathcal{K})\} \cup && (4.21) \\ & \{\neg \mathsf{r}(X,Y) :\!-\ \mathsf{not}\ \mathsf{r}(X,Y), \top(X), \top(Y) \mid r \in N_R^{dl}(\mathcal{K})\} \cup && (4.22) \\ & \{\top(a) \mid a \in N_I^{dl}(\mathcal{K})\}. && (4.23) \end{aligned}$$

Recall, that a rule is *unsafe*, if a variable that occurs in the head does not occur in any positive body literal. The predicate $\top(X)$ ensures safe rules, each of the guessing rules (4.19–4.22) would otherwise be unsafe. This predicate apparently represents the $\top$-concept, to which the statement (4.23) asserts each individual present in $\mathcal{K}$. We show now that $\Pi_{gen}(\mathcal{K})$ computes all $\mathbf{B} \in \mathcal{B}_{\mathcal{K}}$, and each $\mathbf{B} \in \mathcal{B}_{\mathcal{K}}$ determines a solution of $\Pi_{gen}(\mathcal{K})$.

Proposition 4.3 ($\mathcal{B}_{\mathcal{K}} = \mathsf{AS}(\Pi_{gen}(\mathcal{K}))$)**.** *Let* $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$ *be a* $\mathcal{SROIQ}$ *knowledge base and* $\Pi_{gen}(\mathcal{K})$ *the logic program obtained by the translation given in (4.19–4.23). Then, it holds that* $\mathcal{B}_{\mathcal{K}}$ *coincides with the set of all answer sets of* $\Pi_{gen}(\mathcal{K})$.

Proof. Note that any $\mathbf{B} \in \mathcal{B}_{\mathcal{K}}$ does not contain an assertion corresponding to $\top$, since in $\Pi_{gen}(\mathcal{K})$ we introduced the predicate merely to overcome unsafe rules. Nevertheless, we can safely assume that each $\mathbf{B} \in \mathcal{B}_{\mathcal{K}}$ contains $\top(a)$ for each individual occurring in $\mathcal{K}$, in order to show set-equality.[1]

[1] As shown in Chapter 3.2.2, Proposition 3.8, any interpretation $\mathcal{I}$ ultimately maps $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$. Thus, requiring assertions of the form $\top(a)$ for any individual would be without effect, but redundant.

$\mathcal{B}_\mathcal{K} \subseteq \mathsf{AS}(\Pi_{gen}(\mathcal{K}))$ Let $\mathbf{B} \in \mathcal{B}_\mathcal{K}$, we show that $\mathbf{B}$ is an answer set of $\Pi_{gen}(\mathcal{K})$. According to the definition of an answer set, 2.19, that is, (i) $\mathbf{B}$ is a model of $\Pi_{gen}(\mathcal{K})$ iff $\mathbf{B}$ is a model of the reduct $(\Pi_{gen}(\mathcal{K}))^\mathbf{B}$, and (ii) there is no $\hat{\mathbf{B}} \subsetneq \mathbf{B}$ which is also a model of $(\Pi_{gen}(\mathcal{K}))^\mathbf{B}$. Regarding (i), $\mathbf{B}$ must satisfy every rule $\rho \in grnd(\Pi_{gen}(\mathcal{K}))$. We can focus on the rules (4.19–4.22) only, since $\{\top(a) \mid a \in N_I^{dl}(\mathcal{K})\} \subseteq \mathbf{B}$ holds by definition of $\mathbf{B}$. Then for every assertion $\alpha \in \mathbf{B}$ there is a rule $\rho \in grnd(\Pi_{gen}(\mathcal{K}))$, such that $H(\rho) = \alpha$. We distinguish α to be of the form:

> $\alpha = \mathsf{A}(a)$: then $\rho = \mathsf{A}(a) :- \mathsf{not}\ \neg\mathsf{A}(a), \top(a)$, which is satisfied since $\neg\mathsf{A}(a) \notin \mathbf{B}$ and $\top(a) \in \mathbf{B}$.
>
> $\alpha = \neg\mathsf{A}(a)$: then $\rho = \neg\mathsf{A}(a) :- \mathsf{not}\ \mathsf{A}(a), \top(a)$, which is satisfied since $\mathsf{A}(a) \notin \mathbf{B}$ and $\top(a) \in \mathbf{B}$.
>
> $\alpha = \mathsf{r}(a,b)$: then $\rho = \mathsf{r}(a,b) :- \mathsf{not}\ \neg\mathsf{r}(a,b), \top(a), \top(b)$, which is satisfied since $\neg\mathsf{r}(a,b) \notin \mathbf{B}$ and $\top(a), \top(b) \in \mathbf{B}$.
>
> $\alpha = \neg\mathsf{r}(a,b)$: then $\rho = \neg\mathsf{r}(a,b) :- \mathsf{not}\ \mathsf{r}(a,b), \top(a), \top(b)$, which is satisfied since $\mathsf{r}(a,b) \notin \mathbf{B}$ and $\top(a), \top(b) \in \mathbf{B}$.

Regarding subset-minimality (ii), first note that all $\mathbf{B} \in \mathcal{B}_\mathcal{K}$ are of equal size; precisely $|\mathbf{B}| = nm * on^2$, with $n = |N_I^{dl}(\mathcal{K})|$, $m = |N_C^{dl}|$ and $o = |N_R^{dl}(\mathcal{K})|$. Suppose now towards contradiction, there exists $\hat{\mathbf{B}} \subsetneq \mathbf{B}$ and $\hat{\mathbf{B}}$ satisfies $\Pi_{gen}^\mathcal{K}$. Then, for some concept name $A \in N_C^{dl}(\mathcal{K})$ and individual $a \in N_I^{dl}(\mathcal{K})$, we neither find $A(a) \in \hat{\mathbf{B}}$ nor $\neg A(a) \in \hat{\mathbf{B}}$. In consequence, let $\rho_+ = \mathsf{A}(a) :- \mathsf{not}\ \neg\mathsf{A}(a), \top(a)$ and $\rho_- = \neg\mathsf{A}(a) :- \mathsf{not}\ \mathsf{A}(a), \top(a)$ be the ground rules corresponding to $A(a)$ and $\neg A(a)$. Both rules are not satisfied by $\hat{\mathbf{B}}$, which refutes $\hat{\mathbf{B}}$ to be an answer set. The argument is the same for assertions on role names.

$\mathcal{B}_\mathcal{K} \supseteq \mathsf{AS}(\Pi_{gen}(\mathcal{K}))$ Let $\mathcal{I} \in AS(\Pi_{gen}(\mathcal{K}))$ be an answer set of $\Pi_{gen}(\mathcal{K})$. We show that $\mathcal{I} \in \mathcal{B}_\mathcal{K}$. First, we find $\{\top(a) \mid a \in N_I^{dl}(\mathcal{K})\} \subseteq \mathcal{I}$. Then, according to Definition 3.6, it must hold that (i) for each concept name $A \in N_C^{dl}(\mathcal{K})$ and individual $a \in N_I^{dl}(\mathcal{K})$, either $A(a) \in \mathcal{I}$ or $\neg A(a) \in \mathcal{I}$, and (ii) for each role name $r \in N_R^{dl}(\mathcal{K})$ and individuals $a, b \in N_I^{dl}(\mathcal{K})$, either $r(a,b) \in \mathcal{I}$ or $\neg r(a,b) \in \mathcal{I}$. In $grnd(\Pi_{gen}(\mathcal{K}))$, we find $\rho_+ = \mathsf{A}(a) :- \mathsf{not}\ \neg\mathsf{A}(a), \top(a)$ and $\rho_- = \neg\mathsf{A}(a) :- \mathsf{not}\ \mathsf{A}(a), \top(a)$ for each concept name A and individual a. Suppose $\{A(a), \neg A(a)\} \notin \mathcal{I}$, then neither ρ_+ nor ρ_- would be satisfied, and therefore $\mathcal{I}$ would not be an answer set. Therefore, either $A(a) \in \mathcal{I}$, such that ρ_- is satisfied and ρ_+ trivially holds, or $\neg A(a) \in \mathcal{I}$, such that ρ_+ is satisfied. The same argument holds for (ii). □

Proposition 4.3 shows, that $\Pi_{gen}(\mathcal{K})$ generates the complete *bounded interpretation space*. Hence, the problem of deciding satisfiability of the given knowledge base is turned into a search problem. We need to direct the search now by means of defining appropriate constraints ruling out interpretations which not yield bounded models.

4.3.2 Axiom Encoding

Since each possible extension of every named concept and role name used in some knowledge base $\mathcal{K}$, is actually generated by $\Pi_{gen}(\mathcal{K})$, any additional rule constructible over the vocabulary of $\mathcal{K}$ would not contribute any new knowledge (new facts). In consequence, each axiom $\alpha \in \mathcal{R} \cup \mathcal{T}$ is turned into a constraint, ultimately ruling out candidate interpretations. Moreover, each individual assertion in the ABox $\mathcal{A}$ restricts the search space further, since for some present fact $A(a)$ any solution candidate containing $\neg A(a)$ is eliminated.

We will successively introduce appropriate encodings for axioms of each knowledge base component, altogether manifested in the program $\Pi_{chk}(\mathcal{K})$ and will finally show that the program $\Pi(\mathcal{K}) = \Pi_{gen}(\mathcal{K}) \cup \Pi_{chk}(\mathcal{K})$ computes all bounded models of $\mathcal{K}$.

Encoding TBox Axioms

Since $\mathcal{T}$ is normalized, each GCI is of certain form which simplifies the encoding tremendously (cf. Section 4.2). As a result we are able to provide relatively simple constraints, represented by $\Pi_{chk}(\mathcal{T})$, obtained as follows:

$$\Pi_{chk}(\mathcal{T}) \;:=\; \{:\!-\; trans(C_1), \ldots, trans(C_n) \mid \text{for each } \top \sqsubseteq \bigsqcup_{i=1}^{n} C_i \text{ in } \mathcal{T}\} \quad (4.24)$$

Each concept expression C_i is translated according to the function $trans(C_i)$ depicted in Table 4.2. Note, each C_i is only one of the ones given in Definition 4.1, the ones given in the first column; i.e. not complex, with the nice effect of $trans(C_i)$ to be realized non-recursively.

Example 4.4. *Consider the normalized axioms from the Graduate Study Problem, obtained in Example 4.2, denoted as* $\mathcal{T}_{(4.18)}$*:*

$$\begin{aligned}\mathcal{T}_{(4.18)} \;=\; \{\;& \top \sqsubseteq \forall \mathsf{completed}.\neg \mathsf{Q}_1 \;\sqcup\; (\geq 2\ \mathsf{passed}.\mathsf{Q}_2),\ \top \sqsubseteq \neg \mathsf{Q}_2 \sqcup \mathsf{Q}_3,\\ & \top \sqsubseteq \neg \mathsf{Q}_3 \sqcup \mathsf{4HCourse},\ \top \sqsubseteq \neg \mathsf{Q}_3 \sqcup (\leq 1\ \mathsf{offers}^{-}.\mathsf{Q}_4),\ \mathsf{Q}_4(\mathsf{krr}),\ \mathsf{Q}_1(\mathsf{krr}) \quad \}\end{aligned}$$

For which we obtain the following $\Pi_{chk}(\mathcal{T}_{(4.18)})$*:*

$$\begin{aligned}&:\!-\ \mathsf{completed}(X, Y_{Q_1}), \mathsf{Q}_1(Y_{Q_1}), \#\mathsf{count}\{\mathsf{passed}(X, Y_{Q_2})\ :\ \mathsf{Q}_2(Y_{Q_2})\} < 2.\\ &:\!-\ \mathsf{Q}_2(X), \mathsf{not}\ \mathsf{Q}_3(X).\\ &:\!-\ \mathsf{Q}_3(X), \mathsf{not}\ \mathsf{4HCourse}(X).\\ &:\!-\ \mathsf{Q}_3(X), \#\mathsf{count}\{\mathsf{offers}(Y_{Q_4}, X)\ :\ \mathsf{Q}_4(Y_{Q_4})\} > 1.\end{aligned}$$

The obtained assertions $\mathsf{Q}_4(krr)$ and $\mathsf{Q}_1(krr)$ in $\mathcal{T}_{(4.18)}$ are considered to be added to the ABox $\mathcal{A}$, which is treated separately and described in the sequel.

C	$trans(C)$	
A	$\mathsf{not\ A}(X)$	(4.25)
$\neg A$	$\mathsf{A}(X)$	(4.26)
$\{a\}$	$\{\mathsf{not\ O_a}(a)\},\ \{\mathsf{O_a}(a)\}$	(4.27)
$\forall r.A$	$\{\mathsf{not\ A}(Y_A),\ \mathsf{ar}(r,X,Y_A)\}$	(4.28)
$\forall r.\neg A$	$\{\mathsf{ar}(r,X,Y_A),\ \mathsf{A}(Y_A)\}$	(4.29)
$\exists r.Self$	$\mathsf{not\ ar}(r,X,X)$	(4.30)
$\neg\exists r.Self$	$\mathsf{ar}(r,X,X)$	(4.31)
$\geq n\ r.A$	$\#\mathsf{count}\{r(X,Y_A)\ :\ \mathsf{A}(Y_A)\} < n$	(4.32)
$\geq n\ r.\neg A$	$\#\mathsf{count}\{r(X,Y_A)\ :\ \mathsf{not\ A}(Y_A)\} < n$	(4.33)
$\leq n\ r.A$	$\#\mathsf{count}\{r(X,Y_A)\ :\ \mathsf{A}(Y_A)\} > n$	(4.34)
$\leq n\ r.\neg A$	$\#\mathsf{count}\{r(X,Y_A)\ :\ \mathsf{not\ A}(Y_A)\} > n$	(4.35)

Note: O_a is a new concept name unique for a. And $\mathsf{ar}(r,X,Y)$ is defined as follows:

$$\mathsf{ar}(r,X,Y) \ := \begin{cases} r(X,Y) & \text{if } r \text{ is an atomic role} \\ s(Y,X) & \text{if } r \text{ is an inverse role and } r = s^- \end{cases}$$

Table 4.2: Translation of Concept Expressions.

Encoding RBox Axioms

Role assertions and role inclusion axioms are also transformed into constraints, grouped in the program $\Pi_{chk}(\mathcal{R})$. According to their DL semantics, this yields:

$$\begin{aligned}
\Pi_{chk}(\mathcal{R}) \ := \ & \{\text{:-}\ \mathsf{ar}(r,X,Y),\ \mathsf{not\ ar}(s,X,Y) \mid r \sqsubseteq s \in \mathcal{R}\} \cup && (4.36)\\
& \{\text{:-}\ \mathsf{ar}(s,X,Y),\ \mathsf{ar}(r,X,Y) \mid \mathsf{Dis}(r,s) \in \mathcal{R}\} \cup && (4.37)\\
& \{\text{:-}\ \mathsf{ar}(s,X,X) \mid \mathsf{Irr}(s) \in \mathcal{R}\} \cup && (4.38)\\
& \{\text{:-}\ \mathsf{ar}(r,X,Y),\ \mathsf{not\ ar}(r,Y,X) \mid \mathsf{Sym}(r) \in \mathcal{R}\} \cup && (4.39)\\
& \{\text{:-}\ \mathsf{ar}(s,Y,X),\ \mathsf{ar}(s,X,Y) \mid \mathsf{Asy}(r) \in \mathcal{R}\} \cup && (4.40)\\
& \{\text{:-}\ \mathsf{ar}(s,X,Y),\ \mathsf{ar}(s,Y,Z),\ \mathsf{not\ ar}(s,X,Z) \mid \mathsf{Tra}(s) \in \mathcal{R}\} \cup && (4.41)\\
& \{\text{:-}\ \mathsf{ar}(s_1,X,Y_1),\ \ldots,\ \mathsf{ar}(s_n,Y_{n-1},Y_n),\ \mathsf{not\ ar}(r,X,Y_n) \mid && (4.42)\\
& \quad s_1 \circ \cdots \circ s_n \sqsubseteq r \in \mathcal{R}\}.
\end{aligned}$$

We want to stress, that RIAs should be treated in an analogous way as it is done in Section 3.4, and therefore also stripped down to less complex role inclusions of fixed size. We note the corresponding modification to Rule (4.42) as an open issue.

Encoding ABox Axioms

The ABox $\mathcal{A}$ itself represents an input database, which we can directly use. However, it remains to check whether $\mathcal{A}$ does not contain contradictory knowledge; i.e. propositional clashes of the form $\{A(a), \neg A(a)\} \in \mathcal{A}$. Hence, the program $\Pi_{chk}(\mathcal{A})$ consists of $\mathcal{A}$ and one additional constraint for each concept and role name ruling out inconsistent input ABoxes.

$$\Pi_{chk}(\mathcal{A}) \ := \ \mathcal{A} \cup \qquad (4.43)$$

$$\{:\!-\ \mathsf{A}(X),\ \neg\mathsf{A}(X) \mid A \in N_C^{dl}(\mathcal{K})\} \cup \tag{4.44}$$
$$\{:\!-\ \mathsf{r}(X,Y),\ \neg\mathsf{r}(X,Y) \mid r \in N_R^{dl}(\mathcal{K})\}. \tag{4.45}$$

Note that the presence of $\{A(a), \neg A(a)\} \in \mathcal{A}$ does not cause an unsatisfiable program under the answer set semantics, since $\neg$ does not have any meaning under the semantics; $\neg A$ is just another predicate name. Thus, the imposed constraints simulate the known DL semantics.

Theorem 4.5. *Let $\mathcal{K}=(\mathcal{R},\mathcal{T},\mathcal{A})$ be a normalized $\mathcal{SROIQ}$ knowledge base, and $\Pi(\mathcal{K})=\Pi_{gen}(\mathcal{K}) \cup \Pi_{chk}(\mathcal{K})$ be the program obtained by applying Rules (4.19–4.45). Then, it holds:*

$$\mathsf{AS}(\Pi(\mathcal{K})) = \{\mathbf{B} \mid \mathbf{B} \in \mathcal{B}_\mathcal{K} \text{ and } \mathcal{I}_\mathbf{B} \models \mathcal{K}\}$$

Proof. By Proposition 4.3, $\Pi_{gen}(\mathcal{K})$ computes the set $\mathcal{B}_\mathcal{K}$. It remains to show, that $\Pi_{chk}(\mathcal{K})$ obeys the bounded model semantic, and consequently excludes each $\mathbf{B} \in \mathcal{B}_\mathcal{K}$ not inducing a bounded model $\mathcal{I}_\mathbf{B}$.

$\mathsf{AS}(\Pi(\mathcal{K})) \subseteq \{\mathbf{B} \mid \mathbf{B} \in \mathcal{B}_\mathcal{K} \text{ and } \mathcal{I}_\mathbf{B} \models \mathcal{K}\}$ Let $\mathbf{I} \in \mathsf{AS}(\Pi(\mathcal{K}))$ be an answer set of $\Pi(\mathcal{K})$. From Proposition 4.3, we know $\mathbf{I} \in \mathcal{B}_\mathcal{K}$. We show now that the interpretation $\mathcal{I}_\mathbf{I}$ induced by $\mathbf{I}$ is a bounded model of $\mathcal{K}$, and therefore $\mathcal{I}_\mathbf{I} \models \alpha$, for each axiom $\alpha \in \mathcal{K}$. Then, let

$\alpha \in \mathcal{R}$: we distinguish role assertions, and role inclusion axioms:

$\underline{\alpha \in \mathcal{R}_a}$: we show the case $\mathsf{Dis}(r,s)$ and $\mathsf{Asy}(r)$ for role names $r, s \in N_R^{dl}(\mathcal{K})$, as other role assertions are treated analogously. Let $\alpha = \mathsf{Dis}(r,s) \in \mathcal{R}_a$, then by definition of $\Pi_{chk}(\mathcal{R})$, there is a ground constraint $\rho_\alpha = :\!-\ \mathsf{s}(a,b), \mathsf{r}(a,b)$ in $grnd(\Pi_{chk}(\mathcal{R}))$, for all individuals $a, b \in N_I^{dl}(\mathcal{K})$. Since $\mathbf{I}$ is an answer set, $\{\mathsf{s}(a,b), \mathsf{r}(a,b)\} \notin \mathbf{I}$. Consequently either $(a,b) \in \mathsf{s}^{\mathcal{I}_\mathbf{I}}$, or $(a,b) \in \mathsf{r}^{\mathcal{I}_\mathbf{I}}$, hence $\mathcal{I}_\mathbf{I} \models \mathsf{Dis}(r,s)$. Let $\alpha = \mathsf{Asy}(r) \in \mathcal{R}_a$, then by definition of $\Pi_{chk}(\mathcal{R})$, there is a ground constraint $\rho_\alpha = :\!-\ \mathsf{r}(a,b), \mathsf{r}(b,a)$ in $grnd(\Pi_{chk}(\mathcal{R}))$, for all individuals $a, b \in N_I^{dl}(\mathcal{K})$. Since $\mathbf{I}$ is an answer set, $\{\mathsf{r}(a,b), \mathsf{r}(b,a)\} \notin \mathbf{I}$. Consequently either $(a,b) \in \mathsf{r}^{\mathcal{I}_\mathbf{I}}$, or $(b,a) \in \mathsf{r}^{\mathcal{I}_\mathbf{I}}$, hence $\mathcal{I}_\mathbf{B} \models \mathsf{Asy}(r)$.

$\underline{\alpha \in \mathcal{R}_h}$: then let α be of the form $s_1 \circ \ldots \circ s_n \sqsubseteq r$, with $s_1, \ldots, s_n, r \in N_R^{dl}(\mathcal{K})$, and $\rho_\alpha = :\!-\ \mathsf{s_1}(a_1,a_2), \mathsf{s_2}(a_2,a_3), \ldots, \mathsf{s_n}(a_n,a_{n+1}), \mathsf{not}\ \mathsf{r}(a_1,a_{n+1})$ be the ground constraint in $grnd(\Pi_{chk}(\mathcal{R}))$. Since $\mathbf{I}$ is an anser set, we have that, if $\{\mathsf{s_1}(a_1,a_2), \mathsf{s_2}(a_2,a_3), \ldots, \mathsf{s_n}(a_n,a_{n+1})\} \in \mathbf{I}$ implies $\mathsf{r}(a_1,a_{n+1}) \in \mathbf{I}$. And consequently $(a_1,a_2) \in \mathsf{s_1}^{\mathcal{I}_\mathbf{I}}, \ldots, (a_n,a_{n+1}) \in \mathsf{s_n}^{\mathcal{I}_\mathbf{I}}$ and $(a_1,a_{n+1}) \in \mathsf{r}^{\mathcal{I}_\mathbf{I}}$, thus $\mathcal{I}_\mathbf{I} \models s_1 \circ \ldots \circ s_n \sqsubseteq r$.

$\alpha \in \mathcal{T}$: then α is normalized and of the form $\top \sqsubseteq \bigsqcup_{i=1}^{n} C_i$. In Rule (4.24), α is turned into a constraint $\rho_\alpha = :\!-\ trans(C_1), \ldots, trans(C_n)$ in $\Pi_{chk}(\mathcal{T})$. Since $\mathbf{I}$ is an answer set, it does not violate any of the grounded instances of ρ_α in $grnd(\Pi_{chk}(\mathcal{T}))$. Suppose now towards contradiction, $\mathcal{I}_\mathbf{I}$ induced by $\mathbf{I}$ does not satisfy α, $\mathcal{I}_\mathbf{I} \not\models \alpha$. Then, $\mathcal{I}_\mathbf{I} \not\models C_i$, for all $1 \leq i \leq n$. However, since $\mathbf{I}$ does not violate ρ_α, in each of the ground instantiations of ρ_α, there is exists a $trans(C_i)$ which is not satisfied by $\mathbf{I}$, $1 \leq i \leq n$. Then, C_i is one of the expressions given in Definition 4.2, and we distinguish:

$\underline{C_i = A}$: then $trans(C_i) = \mathsf{not}\ A(X)$, and $A(a) \in \mathbf{I}$ for any $a \in N_I^{dl}(\mathcal{K})$. Consequently $a \in A^{\mathcal{I}_\mathbf{I}} \neq \emptyset$, which contradicts the assumption $\mathcal{I}_\mathbf{I} \not\models C_i$.

$\underline{C_i = \neg A}$: then $trans(C_i) = A(X)$, and $\neg A(a) \in \mathbf{I}$ for any $a \in N_I^{dl}(\mathcal{K})$. Consequently $a \in (\neg A)^{\mathcal{I}_\mathbf{I}} \neq \emptyset$, which contradicts the assumption $\mathcal{I}_\mathbf{I} \not\models C_i$.

$\underline{C_i = \{a\}}$: then $trans(C_i) = \{\mathsf{not}\ \mathsf{O_a}(X)\}$ and $\mathsf{O_a}(a)$, thus necessarily $\mathsf{O_a}(a) \in \mathbf{I}$. In order to not satisfy $trans(C_i)$, $X = a$. Consequently we have $a \in O_a^{\mathcal{I}_\mathbf{I}}$ with O_a as nominal guard concept, and therefore $\{a\}^{\mathcal{I}_\mathbf{I}} = \{a\}$, which contradicts the assumption.

$\underline{C_i = \forall r.A}$: then $trans(C_i) = \{r(X, Y_A), \mathsf{not}\ A(Y_a)\}$, and $A(b) \in \mathbf{I}$ whenever $r(a, b) \in \mathbf{I}$. Consequently, $(a, b) \in r^{\mathcal{I}_\mathbf{I}}$ implies $b \in A^{\mathcal{I}_\mathbf{I}}$, which contradicts the assumption $\mathcal{I}_\mathbf{I} \not\models C_i$.

$\underline{C_i = \geq n\, r.A}$: then $trans(C_i) = \#\mathsf{count}\{\mathsf{r}(X, Y_\mathsf{A}) : \mathsf{A}(Y_\mathsf{A})\} < n$, and more or equal than n, say m, atoms $\mathsf{r}(a, b) \in \mathbf{I}$ and $\mathsf{A}(b) \in \mathbf{I}$. Consequently we find also m pairs $(a, b) \in r^{\mathcal{I}_\mathbf{I}}$ and $b \in A^{\mathcal{I}_\mathbf{I}}$, which contradicts the assumption.

All remaining cases can be treated analogously.

$\alpha \in \mathcal{A}$: $\mathbf{I}$ satisfies $\Pi_{chk}(\mathcal{A})$, in particular $\mathcal{A} \subseteq \mathbf{I}$. Moreover, none of the imposed constraints in $\Pi_{chk}(\mathcal{A})$ is violated, proving consistency of $\mathcal{A}$, and therefore $\{\mathsf{A}(a), \neg\mathsf{A}(a)\} \notin \mathbf{I}$ and $\{\mathsf{r}(a, b), \neg\mathsf{r}(a, b)\} \notin \mathbf{I}$ for all concept names $A \in N_C^{dl}(\mathcal{K})$, role names $r \in N_R^{dl}(\mathcal{K})$ and individuals $a, b \in N_I^{dl}(\mathcal{K})$.

$\mathsf{AS}(\Pi(\mathcal{K})) \supseteq \{\mathbf{B} \mid \mathbf{B} \in \mathcal{B}_\mathcal{K}\ and\ \mathcal{I}_\mathbf{B} \models \mathcal{K}\}$ Let $\mathcal{I}_\mathbf{B}$ be a bounded model of $\mathcal{K}$, induced by some $\mathbf{B} \in \mathcal{B}_\mathcal{K}$. We show that $\mathbf{B}$ is an answer set of $\Pi(\mathcal{K}) = \Pi_{gen}(\mathcal{K}) \cup \Pi_{chk}(\mathcal{K})$. From Proposition 4.3 we know, that $\mathbf{B}$ is an answer set of $\Pi_{gen}(\mathcal{K})$, thus it remains to show that $\mathbf{B}$ satisfies $\Pi_{chk}(\mathcal{K})$ and therefore does not violate any of the imposed constraints. Since $\mathcal{I}_\mathbf{B} \models \alpha$, for each $\alpha \in \mathcal{K}$, let

$\alpha \in \mathcal{R}$: we distinguish again role assertions and role chain axioms.

$\alpha \in \mathcal{R}_a$: we show the case $\mathsf{Sym}(r)$ and $\mathsf{Irr}(r)$ for role name $r \in N_R^{dl}(\mathcal{K})$, as other role assertions are treated analogously. Let $\alpha = \mathsf{Sym}(r)$, and $\rho_\alpha = :\!-\ \mathsf{r}(X, Y), \mathsf{not}\ \mathsf{r}(Y, X)$ be the constraint according to Rule (4.39). Since $\mathcal{I}_\mathbf{B} \models \alpha$, we have that if $r(a, b) \in \mathbf{B}$ then $r(b, a) \in \mathbf{B}$, for all $a, b \in N_I^{dl}(\mathcal{K})$. Consequently, ρ_α is not violated. Let $\alpha = \mathsf{Irr}(r)$, and $\rho_\alpha = :\!-\ \mathsf{r}(X, X)$ be the constraint according to Rule (4.38). Since $\mathcal{I}_\mathbf{B} \models \alpha$, $r(a, a) \notin \mathbf{B}$ for all $a \in N_I^{dl}(\mathcal{K})$. Consequently, ρ_α is not violated by $\mathbf{B}$.

$\alpha \in \mathcal{R}_h$: then α is of the form $s_1 \circ \ldots \circ s_n \sqsubseteq r$, with $s_1, \ldots, s_n, r \in N_R^{dl}(\mathcal{K})$, and $\rho_\alpha = :\!-\ \mathsf{s_1}(X, Y_1), \ldots, \mathsf{s_n}(Y_n, Z), \mathsf{not}\ \mathsf{r}(X, Z)$ is the constrained according to Rule (4.42). Since $\mathcal{I}_\mathbf{B} \models \alpha$, for all $a_1, \ldots, a_{n+1} \in N_I^{dl}(\mathcal{K})$ we have that if $s_1(a_1, a_2), \ldots, s_n(a_n, a_{n+1}) \in \mathbf{B}$, then $r(a_1, a_{n+1}) \in \mathbf{B}$. Consequently, ρ_α is not violated by $\mathbf{B}$.

$\alpha \in \mathcal{T}$: then α is normalized and of the form $\top \sqsubseteq \bigsqcup_{i=1}^{n} C_i$, which is satisfied by $\mathcal{I}_\mathbf{B}$, iff $C_i^{\mathcal{I}_\mathbf{B}} \neq \emptyset$ for some $1 \leq i \leq n$. Let $\rho_\alpha = :- \mathit{trans}(C_1), \ldots, \mathit{trans}(C_n)$ be the constraint obtained from α, applying Rule (4.24). We need to show that $\mathbf{B}$ does not violate the constraint. Let C_i be the concept expression for which $C_i^{\mathcal{I}_\mathbf{B}} \neq \emptyset$ holds, $1 \leq i \leq n$, and C_i is one of the expressions given in Definition 4.1, in particular we have:

$\underline{C_i = A}$: then $A^{\mathcal{I}_\mathbf{B}} = \{a \,|\, \mathsf{A}(a) \in \mathbf{B}\}$. Consequently, for each of those $\mathsf{A}(a) \in \mathbf{B}$, $\mathit{trans}(A) = \mathsf{not}\ \mathsf{A}(X)$ is not satisfied.

$\underline{C_i = \neg A}$: then $\neg A^{\mathcal{I}_\mathbf{B}} = \{a \,|\, \neg\mathsf{A}(a) \in \mathbf{B}\}$. Consequently, for each of those $\neg\mathsf{A}(a) \in \mathbf{B}$, $\mathit{trans}(\neg A) = \mathsf{A}(X)$ is not satisfied.

$\underline{C_i = \exists r.\mathit{Self}}$: then $(\exists r.\mathit{Self})^{\mathcal{I}_\mathbf{I}} = \{a \,|\, \mathsf{r}(a,a) \in \mathbf{B}\}$. Consequently, $\mathit{trans}(\exists r.\mathit{Self}) = \mathsf{not}\ \mathsf{r}(X,X)$ is not satisfied for those $\mathsf{r}(a,a) \in \mathbf{B}$.

$\underline{C_i = \leq r\, n.A}$: then $(\leq r\, n.A)^{\mathcal{I}_\mathbf{B}} = \{a \,|\, \#\{\mathsf{r}(a,b) \text{ and } \mathsf{A}(b) \in \mathbf{B}\} = m \leq n\}$. Consequently, $\mathit{trans}(\leq r\, n.A) = \#\mathsf{count}\{\mathsf{r}(X,Y_A) : \mathsf{A}(Y_A)\} > n$, is not satisfied since there are m such $\mathsf{r}(a,b) \in \mathbf{B}$ with $\mathsf{A}(b) \in \mathbf{B}$.

All remaining cases can be treated analogously.

$\alpha \in \mathcal{A}$: then $\alpha \in \mathbf{B}$, as well as $\alpha \in \Pi_{chk}(\mathcal{A})$, since by definition $\mathcal{A} \in \Pi_{chk}(\mathcal{A})$. In general, since $\mathcal{I}_\mathbf{B} \models \mathcal{K}$, $\mathcal{A}$ is consistent and therefore $\{\mathsf{A}(a), \neg\mathsf{A}(a)\} \notin \mathbf{B}$, as well as $\{\mathsf{r}(a,b), \neg\mathsf{r}(a,b)\} \notin \mathbf{B}$ for all concept names A, role name r and individuals a, b.

□

4.4 Summary

For any normalized knowledge base $\mathcal{K}$, we developed an encoding into an answer set program $\Pi(\mathcal{K})$. With the proposed *naïve translation* we simply stick to the introduced bounded model semantics in a declarative fashion. That is, we first encode how potential bounded models look like and do so via the illustrated alternative characterization; i.e. generate potential ground models. Then, each RBox and TBox axiom is turned into an adequate constraint, all ruling out potential interpretations not inducing bounded models. Theorem 4.3, shows that the set of bounded models of $\mathcal{K}$ coincides with the set of answer sets $\mathsf{AS}(\Pi(\mathcal{K}))$. We benefit from this approach in many aspects. Most notably, the imposed reasoning tasks *model extraction* and *model enumeration* are realized without additional efforts, since both are natural tasks for answer set solvers. Moreover, we can realize a decision procedure for the $\mathsf{SAT}_{\mathsf{bm}}$problem; Theorem 4.3 also can be seen as sound and complete computation of bounded models, whereas the non-existence of function symbols guarantees termination.

5 Wolpertinger: A Bounded Model Reasoner for the Semantic Web

Based on the elaborated results, we will present now Wolpertinger, a bounded model reasoner and lightweight implementation of our proposed approach. After describing its system architecture, including all main components and their interdependencies, we conduct an initial evaluation which underpin our theoretical findings and in general leaves a promising impression.

5.1 Overview

Figure 5.1 depicts an architectural overview, emphasizing the main components and their interdependencies. We use a class diagram like notation, merely to show which component makes use of the other, and is realizing which interfaces. We neglect details and remain on an abstract level within this elaboration.

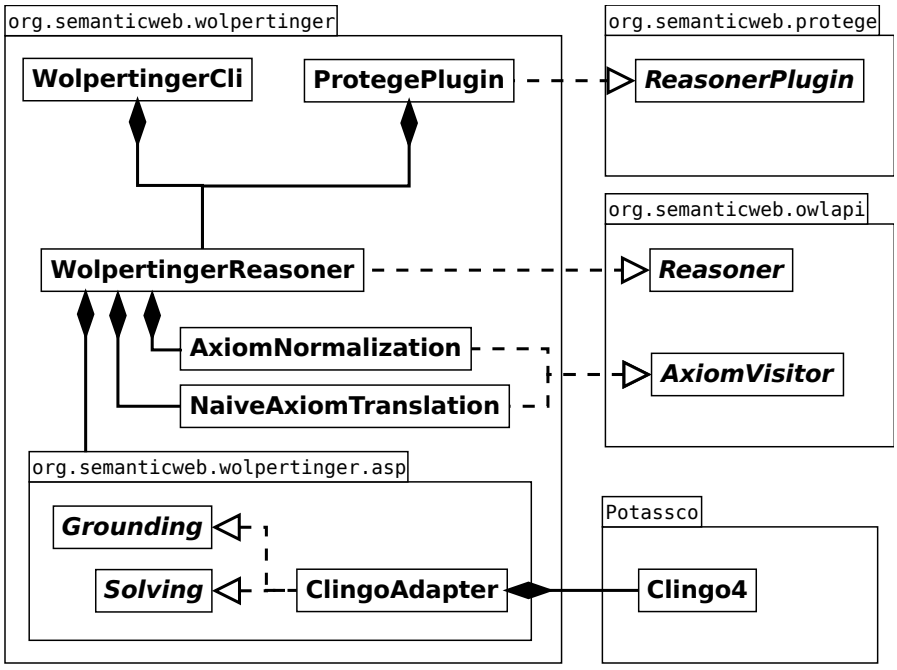


Figure 5.1: Architectural Overview on Component Level.

We ultimately intend to put Wolpertinger into practice as open-source project. However, at the time of finishing this thesis work, not all components were developed thoroughly. We therefore also want to stress, that the sketched design has prototypical character and merely shall serve to underpin our theoretical findings.

5.2 Component Descriptions

Each component is described subsequently, where the order directly represents the data flow and interaction sequence at runtime.

5.2.1 Component: `WolpertingerCommandLine`

As a user interface, the `WolpertingerCommandLine` was designed to initialize the reasoner (`WolpertingerReasoner`) with concrete configuration parameters, based on a user's input. Most notably, these parameters specify the reasoning task of interest, debug instructions, output options, and physical location of the knowledge base of interest.

5.2.2 Component: `ProtegePlugin`

Alternatively, the `ReasonerPlugin` provided by the Protege framework is implemented, such that our reasoner can be used from within the Protégé ontology editor. This component, however, was not developed until this thesis document was finished, since it is simple not needed to conduct a first evaluation.

5.2.3 Component: `WolpertingerReasoner`

The `WolpertingerReasoner` represents the central component orchestrating all phases involved in our approach, which in sequence is

1. Loading the requested knowledge base,
2. Normalize the knowledge base,
3. Translate the knowledge base, and
4. Interact with the ASP solver and therewith perform the requested reasoning task.

Loading the requested knowledge base merely is done via the provided functionality of the OWL API, ultimately resulting in an in-memory representation of the knowledge base as `OWLOntology` object. Remaining steps 2.–4. are delegated to corresponding components and described separately.

5.2.4 Component: `AxiomNormalization`

The imposed normalization described in Section 4.2, is realized in the `AxiomNormalization` component. Conveniently, this is an implementation of the `AxiomVisitor` interface provided by the OWL API, which allows an implementation directly corresponding to the specified normalization in Table 4.1, Section 4.2. After normalizing, the knowledge base is kept in a list of arrays, each array representing a normalized axiom of the form $\top \sqsubseteq \bigsqcup_{i=1}^{n} C_i$, such that the array contains n `OWLAxiom` objects corresponding to one of the disjuncts C_i, $1 \leq i \leq n$.

5.2.5 Component: `NaiveAxiomTranslation`

Our proposed translation is realized within the `NaiveAxiomTranslation` component, which itself is again an implementation of the `AxiomVisitor` interface. This admits a quite generic treatment of each disjunct (`OWLAxiom`) of a normalized axiom, as it is specified in Table 4.2, Section 4.3.2. The resulting logic program complies to the ASP-Core-2 syntax (cf. Section 2.2.1). We keep track of concept and role names and their syntactic counterpart predicate names via a mapping, allowing retranslation of an answer set into a proper ABox.

In this implementation the full logic program is kept in a plain string representation. The reason therefore is mainly that there is no need for intermediate modifications while reasoning, and essentially, the Clingo solver used to compute answer sets anyway accepts logic programs as plain text only.

5.2.6 Component: `ClingoAdapter`

We have chosen to use Clingo from the Potassco[1] suite, as to our knowledge it is the only implementation already supporting the ASP-Core-2 language, and besides, is continuously top-ranked in benchmark evaluations.[2] In order to incorporate the external Clingo tool, the `ClingoAdapter` represents a bridging component offering basic solving functionality while actually calling the external solver.

Mainly the constructed logic program and configuration parameters are passed to Clingo, which is executed in a separate thread. Results are awaited via an established `PipedInputStream` connection, and therefore directly parsed from the textual output of Clingo. Each answer set is parsed and then represented as set of `OWLIndividualAxiom` objects.

5.3 Evaluation

In this section, the implemented approach is evaluated by means of runtime tests on some particular problem instances. Note that we will not provide an intensive benchmark or comparison of our proposed ASP encoding against other approaches, i.e. constraint

[1] Potsdam Answer Set Solving Collection (http://potassco.sourceforge.net).
[2] An overview can be found on http://potassco.sourceforge.net/trophy.html.

solvers. Moreover we do also not compare popular existing OWL and OWL 2 reasoners how they perform on bounded model reasoning.

We used HermiT in its current version 1.3.8. The given runtimes are those given be the tools itself, whereas for both tools only the grounding + solving time, respectively reasoning time, is considered; i.e. without translating the knowledge base, or loading and normalizing the knowledge base in HermiTs case.

5.3.1 Unsatisfiability

Reconsidering the unsatisfiable knowledge base depicted in Example 3.14, Section 3.5, which enforces an unavoidable exploration of the full search space. We can also not hope that our approach is able to solve large instances in reasonable time. Table 5.1 obeys runtimes of Wolpertinger and HermiT for detecting unsatisfiability, with considerably small instances. HermiT is working on input knowledge bases using the $\top$-restriction workaround. Whereas Wolpertinger takes the same knowledge base without $\top$-restriction as input. We carefully identify, that in general we detect unsatisfiability faster. Whereas in the last run, we can still claim unsatisfiability in 85 seconds, HermiT was beyond 10 minutes when it was stopped. We provided the number of backtracking points needed in HermiTs case, to emphasize the underlying combinatorial problem.

#	Nominals	Concepts	Wolpertinger	HermiT	#Backtrackings
1	5	6	$< 1\,ms$	$148\,ms$	260
2	6	7	$< 1\,ms$	$436\,ms$	1630
3	7	8	$40\,ms$	$1050\,ms$	11742
4	8	9	$320\,ms$	$2292\,ms$	95900
5	9	10	$4790\,ms$	$15301\,ms$	876808
6	10	11	$54400\,ms$	$144024\,ms$	8877690
7	11	12	$85080\,ms$	/	$> 50 \times 10^6$

Table 5.1: Runtime comparison for detecting unsatisfiability, Example 3.5.

5.3.2 Model Extraction and Model Enumeration

Focusing on the main use cases our approach is actually motivated with, we continue in providing some figures for model extraction; i.e. satisfiability witnessed by some bounded model. As well as model enumeration; i.e. requesting all bounded models of the given input knowledge base. We use the widely known Sudoku riddle, which modeled in OWL has a total number of 189 named individuals, 14 concept names as well as 2 role names, and therefore represents already an adequately problem instance comprising fair enough individuals. We will now not compare runtimes of Wolpertinger and HermiT, since the imposed problem is already to large to handle for HermiT, at least with standard settings on our desktop machine.

In average, we can solve a given Sudoku instance in 40 seconds, of which actually 24 seconds account to grounding and around 15 seconds for preprocessing, and therefore less than one second for the actual solving. However, instead of solving different Sudoku instances and hence searching for one model, we want to demonstrate the advanced *enumeration* task and simply generate Sudoku instances instead of solving them; i.e. we consider an empty Sudoku board, consequently searching for bounded models representing a proper Sudoku instance according to the riddle's constraints.

#	**Models Requested**	**Time (Total)**	**Time (Solving)**
1	100	40.989 s	< 0.01 s
2	1000	41.354 s	0.11 s
3	10000	45.533 s	1.16 s
4	100000	61.759s	11.01 s
5	1000000	157.310 s	110.00 s
n	...	...	...

Table 5.2: Enumerating Sudoku Instances – Runtime Overview.

Table 5.2 shows vividly, that for instances of this size, the total time accumulates mainly in the required grounding time. In this case, the size of the grounded program is 160 MByte. Clearly, in this setup the pure solving time grows linear with number of requested models. We therefore did not request more, not to mention requiring all models to be enumerated, which would be ridiculously.

6 Conclusions, Related and Future Work

In this thesis we considered an modification of the standard DL semantics, which we referred to as *bounded model semantics*. We argue that in certain application domains, bounded models represent more adequate what is expected as a model. That is, the domain is composed only by named individuals explicitly given in the knowledge base of interest. From an ontology engineering view point, this intuitively means that models are expected to be constructed over these individuals only, not more and not less. This restriction actually limits the expressive power of $\mathcal{SROIQ}$, since henceforth only finite domains are viable, leading to NP-completeness of deciding satisfiability (cf. Theorem 3.13, Section 3.4). We demonstrated in Section 3.5 the possibility to enforce bounded models using an additional axiom, such that existing reasoners can be employed. However, this workaround introduces an additional and potentially large non-deterministic choice, resulting in potentially many backtracking points (cf. Example 3.14).

Therefore, in Chapter 4, a novel approach is proposed for reasoning with respect to bounded models. We compute bounded models of a given knowledge base $\mathcal{K}$, by encoding the knowledge base itself into an *answer set program* $\Pi(\mathcal{K})$ in such a way, that each answer set of $\Pi(\mathcal{K})$ is a bounded model of $\mathcal{K}$, and likewise each bounded model is an answer set (cf. Theorem 4.5). This rather strong relation allows us not only to test satisfiability w.r.t. to the bounded model semantics, but consequently reduce all reasoning tasks to satisfiability, respectively unsatisfiability. One could state, that we treat the satisfiability problem as directed search problem under constraints; i.e. a *constraint satisfaction problem*.

In Chapter 5, we sketch the architectural design of a reasoner named Wolpertinger. An OWL 2 compatible implementation based on the proposed translation. Based on an empirical evaluation, we can generally claim to perform faster in satisfiability and unsatisfiability checking than HermiT, the dominating $\mathcal{SROIQ}$ reasoner. Regarding model enumeration, which we could not compare to other approaches, the evaluation supports the argument, that the proposed naïve encoding already yields an acceptable runtime performance (cf. Section 5.3).

In conclusion, we provide an elegant, although unusual, approach towards bounded model reasoning and its imposed non-standard reasoning tasks *model extraction* and *model enumeration*. Moreover, we also contribute to the field of logic programming, in particular answer set programming, by means of an interesting use case and therefore bringing together two allegedly different knowledge representation formalisms in a benefiting setup; sort of a win-win situation. Instead arguing from the semantic web perspective, we can very well perceive this interplay from the logic programming side, and

stress that semantic web technologies, especially OWL and the ontology editor Protégé[1], can be used as high-level tools for modeling certain types of constraint problems. Our approach then provides an automated encoding of the modeled problem into an answer set program, with the advantage that an untrained user is not in touch with the actual logic program. We will discuss this aspect further in the remaining sections.

6.1 Related Work

First, we relate our work only with respect to the proposed semantics, and therefore to other approaches dealing with finite models. Most notably, this refers to so-called *finite model reasoning*, a field not only of interest within the DL communities, where one is interested in models of any size, but finite. With respect to DLs, for example the work of Lutz et al. [LST05], or Rosati [Ros08] with respect to DL-lite. However, it clearly differs from our interests, since we are looking into models of a priori known size.

Continuing with *model enumeration*, to the best of our knowledge, this task is not generally addressed within the field of DL by any other approach so far.

In our work, logic programming is used as an alternative computational method. Thus, we do not try to enhance expressivity with non-monotonic or other features. Closely related therefore, is the work of Motik [Mot06], reducing $\mathcal{SHIQ}(\mathbf{D})$ knowledge bases into *disjunctive datalog programs* [EGM97], motivated in efficient reasoning with large quantities of assertional knowledge. Main result of Motiks work, which distinguishes from our, is an encoding, such that the same set of ground atoms is entailed by the resulting datalog program as it is by the given knowledge base. Huge efforts have been put in extending DLs with non-monotonic features; for example, the ability to work with default knowledge, or express exceptions. However, all these extension only relate to our approach insofar, that DLs and logic programming are combined in some way, almost always to enhance and add expressive features. For example, Grosof et al. [GHVD03], introduced the notion of *description logic programs* (DL-programs), continued by Eiter et al. in [EIL+08], which consists of a DL knowledge base and a finite set of DL-rules. DL-programs under the answer set semantics can be used to realize different forms of closed-world reasoning [EIL+08].

6.2 Future Research

The work in this thesis can be seen as the first fundamental bricks, bridging two knowledge representation formalisms in a fertile way. We will outline briefly potentially future work, some of it merely represents ideas which might be worth to consider, whereas others are concrete improvements or simply open issues.

[1] http://protege.stanford.edu

6.2.1 Optimized Translation

First, our translation so far does not enjoy any optimization and thus is said to be naïve in its nature. Certainly there is huge potential for optimization in many aspects. There is the normalization, mainly performing the structural transformation as it is performed in [MSH09]. Although we need to normalize axioms and break complex axiom into simpler ones, it is not alway necessary to introduce auxiliary concept names, as we can in many cases encode complex concepts directly into corresponding rules. For example, consider $A \sqsubseteq \exists r.(B_1 \sqcup \neg B_2)$, which can be directly encoded as the constraint $:- \mathsf{A}(X), \#\mathsf{count}\{\mathsf{r}(X,Y) \ : \ \mathsf{not}\ \mathsf{B_1}(Y), \mathsf{not}\ \neg\mathsf{B_2}(Y)\} < 1$. There are several such cases which would need to be inspected. Also, guessing extensions can in some cases be more directed, for example, when domain and range axioms are defined for some role, it is not necessary to guess blindly, but rather over the provide range and domain concepts.

6.2.2 Justifying Unsatisfiability

Naturally, some knowledge base might be unsatisfiable for non-trivial reasons. With respect to the naïve translation proposed, one or more of the encoded knowledge base axioms are violated not permitting any answer set. We experimented already with slightly different encodings of axioms, allowing constraints to be activated or deactivated by simply introducing new body atoms unique for each constraint. Then, we also guess which constraint is actually activated and observer whether it affects the satisfiability result. Using the #max-aggregate function, we can ask for satisfiability having a maximal number of constraints activated. A simple pinpointing then consists of identifying those constraints which are not activated. However, without preceding optimization efforts we can not hope for reasonable performance.

6.2.3 Making use of non-monotonic features

Our encoding actually does not really use non-monotonic features. In this work we intended to keep the proposed bounded model semantics monotonic. However, based on some experiments, we could also consider something like *minimal bounded models*. This would enable an entirely different translation, more strongly related to answer sets and probably even more tailored to constraint problems.

6.2.4 Open Issues

– **Data Types**. OWL 2 DL is actually based on $\mathcal{SROIQ}(\mathbf{D})$, which is an extension of $\mathcal{SROIQ}$ with concrete domains. It should be possible to extend our encoding, in order to fully take data types into account.

Bibliography

[BCM+07] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, second edition, 2007.

[BS01] Franz Baader and Ulrike Sattler. An overview of tableau algorithms for description logics. *Studia Logica*, 69(1):5–40, 2001.

[Cal96] Diego Calvanese. Finite model reasoning in description logics. *Proceedings of KR*, 96:292–303, 1996.

[CFG+13] Francesco Calimeri, Wolfgang Faber, Martin Gebser, Giovambattista Ianni, Roland Kaminski, Thomas Krennwallner, Nicola Leone, Francesco Ricca, and Torsten Schaub. Asp-core-2 input language format. 2013.

[EGM97] Thomas Eiter, Georg Gottlob, and Heikki Mannila. Disjunctive Datalog. *ACM Trans. Database Syst.*, 22(3):364–418, 1997.

[EIL+08] Thomas Eiter, Giovambattista Ianni, Thomas Lukasiewicz, Roman Schindlauer, and Hans Tompits. Combining answer set programming with description logics for the Semantic Web. *Artificial Intelligence*, 172:1495–1539, 2008.

[GHM+13] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. Hermit: an owl 2 reasoner. *Journal of Automated Reasoning*, pages 1–25, 2013.

[GHVD03] Benjamin N Grosof, Ian R Horrocks, Raphael Volz, and Stefan Decker. Description Logic Programs : Combining Logic Programs with Description Logic. In *WWW '03 Proceedings of the 12th international conference on World Wide Web*, pages 48–57, 2003.

[GKKS12] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. Answer Set Solving in Practice. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6:1–238, 2012.

[HKS06] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The Even More Irresistible $\mathcal{SROIQ}$. *Proceedings of KR*, 6:57–67, 2006.

[Kaz08] Yevgeny Kazakov. $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are harder than $\mathcal{SHOIQ}$. 2008.

[Lif02] Vladimir Lifschitz. Answer set programming and plan generation. *Artificial Intelligence*, 138(1):39–54, 2002.

[LST05] Carsten Lutz, Ulrike Sattler, and Lidia Tendera. The complexity of finite model reasoning in description logics. *Information and Computation*, 199(1-2):132–171, May 2005.

[Mot06] Boris Motik. *Reasoning in Description Logics using Resolution and Deductive Databases.* PhD thesis, Karlsruhe Institute of Technology, 2006.

[MSH09] Boris Motik, Rob Shearer, and Ian Horrocks. Hypertableau Reasoning for Description Logics. *Artificial Intelligence Research*, 36:165–228, 2009.

[PG86] David A Plaisted and Steven Greenbaum. A structure-preserving clause form translation. *Journal of Symbolic Computation*, 2(3):293–304, 1986.

[Ros08] Riccardo Rosati. Finite model reasoning in dl-lite. In *The Semantic Web: Research and Applications: 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain*, volume 5021, page 215. Springer, 2008.

[Rud11] Sebastian Rudolph. Foundations of Description Logics. In Axel Polleres, Claudia D'Amato, Marcelo Arenas, Siegfried Handschuh, Paula Kroner, Sascha Ossowski, and Peter F Patel-Schneider, editors, *Reasoning Web*, volume 6848 of *Lecture Notes in Computer Science*, pages 76–136. Springer, 2011.

[Sat07] Ulrike Sattler. *Reasoning in description logics: Basics, extensions, and relatives*, volume 4636 of *Lecture Notes in Computer Science*. 2007.

[Sir06] Evren Sirin. From wine to water: Optimizing description logic reasoning for nominals. In *Proceedings of KR-2006.(2006) 90–99*, 2006.

[W3C09] W3C OWL Working Group. *OWL˜2 Web Ontology Language: Document Overview*. W3C Recommendation, 2009.