Fixed-Domain Reasoning for Description Logics

Sarah Gaggl, Sebastian Rudolph, Lukas Schweizer
Technische Universität Dresden
Computational Logic Group
firstname.lastname@tu-dresden.de

Abstract. Description logics (DLs) are currently a de facto standard in logic-based knowledge representation thanks to the last two decades of research and their use as the underpinning of standardized and widely adopted web ontology language (OWL), which also comes with the advantage of existing user-friendly modeling tools. It has, however, been observed that OWL and description logics are utilized by logically less skilled practitioners as constraint languages adopting a closed-world assumption, contrary to the open world semantics imposed by the classical definitions and the standards. Therefore, we came up with an alternative formal semantics reflecting this "off-label use" of these widely adopted formalisms. To that end, we introduce the *fixed-domain semantics* and discuss that this semantics gives rise to the interesting new inferencing task of *model enumeration*. We describe how the new semantics can be axiomatized in very expressive DLs. We thoroughly investigate the complexities for standard reasoning as well as query answering for a wide range of DLs. We present an implementation of a fixed-domain DL reasoner based on the translation into answer set programming (ASP) and provide first results that this tool is superior to alternative approaches when used on constraint-satisfaction-type problems.

1 Introduction

Preferred knowledge representation formalisms are often the ones which are standardized, widely adopted and come with elaborate modeling tool support. One outstanding example for this is certainly the Web Ontology Language OWL [32]. Ontology editors like Protégé [16] provide user-friendly interfaces and combined with the natural-language-like Manchester syntax [12] possess perspicuous access to an arguably complex and involved formalism.

This gives rise to scenarios in which OWL is chosen over other formalisms, even if the application scenario does not match the typical usage of this language. For example, the modeled problem might be of a constraint-satisfaction type which does not go well with OWL's standard semantics allowing for models of arbitrary size. Consider the 3-coloring problem as a short but representative constraint-satisfaction problem, for which one can easily envision some OWL axioms imposing the conditions on valid colorings of a given graph. Then, asking for consistency of the problem description is a natural task for OWL reasoners,

asking for colorability as such can be cast into a satisfiability problem, but asking for concrete colorings already requires reasoning capabilities none of the OWL reasoners we are aware of is furnished with.

To overcome these shortcomings, we propose *fixed-domain reasoning* for description logics – a family of logics providing the logical underpinning of OWL and its sublanguages. By this intuitive and simple approach, we consider DLs under a non-standard model-theoretic semantics, modifying the modelhood condition by restricting the domain to an explicitly given fixed finite set. We investigate the combined complexity of reasoning in the presence of a given fixed domain for a wide range of description logics, for which we establish tight bounds for standard reasoning tasks as well as query answering for various query notions. While satisfiability checking in OWL under the classical semantics is N2ExpTime-complete [15] and query answering is not even known to be decidable, we show that these problems under the *fixed-domain semantics* are merely NP-complete and Π_2^{P}-complete, respectively.

We note that the fixed-domain condition can be axiomatized in OWL. Still, employing the axiomatization, existing OWL reasoners struggle on fixed-domain reasoning, due to the heavy combinatorics involved. Therefore, we propose a different approach and define a translation of $\mathcal{SROIQ}$ knowledge bases (the logical counterparts to OWL ontologies) into answer set programming (ASP) [4], such that the set of (fixed-domain) models coincides with the set of answer-sets of the obtained program. This allows us to use existing ASP solvers (see [5] for an overview) for fixed-domain reasoning – including standard as well as non-standard tasks. For the proposed translation, we provide an implementation and present preliminary evaluations on typical constraint-satisfaction-type problems. This not only demonstrates feasibility, but also suggests significant improvement compared to the axiomatized approach using highly optimized OWL reasoners.

2 Preliminaries

We assume the reader to be familiar with the basics of description logics (DLs) [2, 26]. Nevertheless, we recall some basics of the description logic $\mathcal{SROIQ}$ as well as the class of queries we consider in this work.

OWL 2 DL, the version of the Web Ontology Language we focus on, is defined based on the description logic $\mathcal{SROIQ}$ (for details see [13]). Let N_I, N_C, and N_R be finite, disjoint sets called *individual names*, *concept names* and *role names* respectively. These atomic entities can be used to form complex ones. A $\mathcal{SROIQ}$ *knowledge base* is a tuple $(\mathcal{A}, \mathcal{T}, \mathcal{R})$ where $\mathcal{A}$ is a $\mathcal{SROIQ}$ ABox, $\mathcal{T}$ is a $\mathcal{SROIQ}$ TBox and $\mathcal{R}$ is a $\mathcal{SROIQ}$ RBox, and we will refer to each TBox axiom as *general concept inclusion* (GCI). The semantics of $\mathcal{SROIQ}$ is defined via interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ composed of a non-empty set $\Delta^{\mathcal{I}}$ called the *domain of* $\mathcal{I}$ and a function $\cdot^{\mathcal{I}}$ mapping individual names to elements of $\Delta^{\mathcal{I}}$, concept names to subsets of $\Delta^{\mathcal{I}}$ and role names to subsets of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. This mapping is extended to complex role and concept expressions and finally used to define satisfaction of axioms (for details see [13]). We say that $\mathcal{I}$ satisfies a

knowledge base $\mathcal{K} = (\mathcal{A}, \mathcal{T}, \mathcal{R})$ (or $\mathcal{I}$ is a model of $\mathcal{K}$, written: $\mathcal{I} \models \mathcal{K}$), if it satisfies all axioms of $\mathcal{A}$, $\mathcal{T}$, and $\mathcal{R}$. We say that a knowledge base $\mathcal{K}$ *entails* an axiom α (written $\mathcal{K} \models \alpha$) if all models of $\mathcal{K}$ are models of α.

Boolean Datalog Queries Here we briefly introduce syntax and semantics of Datalog queries over description logic knowledge bases. A *term* can be a variable from a countably infinite set V of variables, or an element of N_I. An *atom* has the form $p(t_1, \ldots, t_n)$ where $t_1, \ldots, t_n$ are terms and p is a predicate of arity n from a set Π of predicates containing N_C (arity 1) and N_R (arity 2) and containing a special predicate *goal* of arity 0. A *Boolean Datalog query* is a set of first order logic Horn rules of the form $\forall X.a_1 \wedge \ldots \wedge a_k \rightarrow a$ where $a_1, \ldots, a_n, a$ are atoms, but the predicate of a is not from N_C or N_R. $X \subseteq V$ denotes the set of variables occurring in the atoms. Given a DL interpretation $\mathcal{I}$, and a Boolean Datalog query Q, an *extended model* for $\mathcal{I}$ and Q is a first-order interpretation $\mathcal{J}$ over $\Delta^{\mathcal{I}}$ that coincides with $\mathcal{I}$ on the interpretation of N_C and N_R and satisfies all the rules from Q. We say that Q *matches* $\mathcal{I}$ and write $\mathcal{I} \models Q$ if $\mathcal{J} \models$ *goal* for every extended model $\mathcal{J}$ for $\mathcal{I}$ and Q. For a DL knowledge base $\mathcal{K}$, we say $\mathcal{K}$ *entails* Q iff $\mathcal{I} \models Q$ for every model $\mathcal{I}$ of $\mathcal{K}$. *Bounded arity Datalog queries* are classes of queries where the arity of the used predicates is bounded by some constant. A *Boolean conjunctive query* is a Boolean Datalog query with just one rule where $a_1, \ldots, a_n$ use only predicates from $N_C \cup N_R$ and $a =$ *goal*. In that case, such a query can be equivalently written as the first-order formula $\exists X.a_1 \wedge \ldots \wedge a_k$.

3 Models over Fixed Domains

In DLs, models can be of arbitrary cardinality. In many applications, however, the domain of interest is known to be finite. In fact, restricting reasoning to models of finite domain size (called *finite model reasoning*, a natural assumption in database theory), has already become the focus of intense studies in DLs [17, 6, 25, 27]. As opposed to assuming the domain to be merely finite (but of arbitrary, unknown size), we consider the case where the domain has an *a priori known cardinality* and use the term *fixed domain*. We refer to such models as *fixed-domain models* and argue that in many applications, this modification of the standard DL semantics represents a more intuitive definition of what is considered and expected as a *model* of some knowledge base.

Definition 1 (Fixed-Domain Semantics). *Given a DL knowledge base $\mathcal{K}$ and a non-empty finite set $\Delta_{\mathcal{K}} \subseteq N_I$, called* fixed domain*, an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is said to be $\Delta_{\mathcal{K}}$-*fixed *(or just* fixed*, if $\Delta_{\mathcal{K}}$ is clear from the context), if $\Delta^{\mathcal{I}} = \Delta_{\mathcal{K}}$ and $a^{\mathcal{I}} = a$ for all $a \in \Delta_{\mathcal{K}}$. Accordingly, we call an interpretation $\mathcal{I}$ a $\Delta_{\mathcal{K}}$-*model *of $\mathcal{K}$, if $\mathcal{I}$ is a $\Delta_{\mathcal{K}}$-fixed interpretation and $\mathcal{I} \models \mathcal{K}$. A knowledge base $\mathcal{K}$ is called $\Delta_{\mathcal{K}}$-*satisfiable *if it has a model over $\Delta_{\mathcal{K}}$. We say $\mathcal{K}$ $\Delta_{\mathcal{K}}$-*entails *an axiom α ($\mathcal{K} \models_{\mathrm{fd}} \alpha$) if every $\Delta_{\mathcal{K}}$-model of $\mathcal{K}$ is also a model of α.*

Note that, under the fixed-domain semantics, there is a one-to-one correspondence between $\Delta_{\mathcal{K}}$-interpretations and sets of ground facts. That is, for every $\Delta_{\mathcal{K}}$-interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, we find exactly one ABox $\mathcal{A}_{\mathcal{I}}$ with atomic

concept assertions and role assertions defined by $\mathcal{A}_\mathcal{I} := \{r(a,b) \mid (a,b) \in r^\mathcal{I}\} \cup \{A(a) \mid a \in A^\mathcal{I}\} \cup \{\mathit{Ind}_c(a) \mid c \in N_I(\mathcal{K}) \setminus \Delta_\mathcal{K} \text{ and } c^\mathcal{I} = a\}$ and likewise, every such ABox $\mathcal{A}$ gives rise to a corresponding interpretation $\mathcal{I}_\mathcal{A}$.[1] This allows us to use ABoxes as convenient representations of models.

Example 1. We briefly demonstrate the effects of the fixed-domain semantics as opposed to the finite-model semantics (with entailment $\models_{\text{fin}}$) and the classical semantics. Let $\mathcal{K}=(\mathcal{A},\mathcal{T},\mathcal{R})$ and $\Delta_\mathcal{K} = \{a,b\}$ with $\mathcal{A} = \{A(a), A(b), s(a,b)\}$, $\mathcal{T} = \{\top \sqsubseteq \exists r.B, \top \sqsubseteq \leqslant 1\, r^-.\top\}$, and $\mathcal{R}=\{\mathsf{Dis}(s,r)\}$. First we note that $\mathcal{K}$ has a $\Delta_\mathcal{K}$-model $\mathcal{I}$ representable as $\mathcal{A}_\mathcal{I} = \{A(a), A(b), B(a), B(b),\ s(a,b), r(a,a), r(b,b)\}$, thus $\mathcal{K}$ is satisfiable under all three semantics. Then $\alpha = \top \sqsubseteq \exists r.\exists r.B$ holds in all models of $\mathcal{K}$, therefore $\mathcal{K} \models \alpha$, $\mathcal{K} \models_{\text{fin}} \alpha$, and $\mathcal{K} \models_{\text{fd}} \alpha$. Opposed to this, $\beta = \top \sqsubseteq B$ merely holds in all finite models, whence $\mathcal{K} \models_{\text{fin}} \beta$ and $\mathcal{K} \models_{\text{fd}} \beta$, but $\mathcal{K} \not\models \beta$. Finally, $\gamma = \top \sqsubseteq \exists r.\mathit{Self}$ only holds in all $\Delta_\mathcal{K}$-models, thus $\mathcal{K} \models_{\text{fd}} \gamma$, but $\mathcal{K} \not\models_{\text{fin}} \gamma$ and $\mathcal{K} \not\models \gamma$.

Extraction & Enumeration of $\Delta_\mathcal{K}$-Models When performing knowledge base satisfiability checking in DLs (the primary reasoning task usually considered), a model constructed by a reasoner merely serves as witness to claim satisfiability, rather than as an accessible artifact. However, as mentioned before, our approach aims at scenarios where a knowledge base is a formal problem description for which each model represents one solution; in particular the domain is part of the problem description. Then, retrieval of one, several, or all models is a natural task, as opposed to merely checking model existence. With *model extraction* we denote the task of materializing an identified model in order to be able to work with it, i.e. to inspect it in full detail and reuse it in downstream processes. The natural continuation of model extraction is to make all models explicit, performing *model enumeration*. Conveniently, for both tasks, we can use the introduced model representation via ABoxes. Most existing DL reasoning algorithms attempt to successively construct a model representation of a given knowledge base. However, most of the existing tableaux reasoners do not reveal the constructed model, besides the fact that models might end up being infinite such that an explicit representation is impossible. Regarding enumeration, we state that this task is not supported – not even implicitly – by any state-of-the-art DL reasoner, also due to the reason that in the standard case, the number of models is typically infinite and often even uncountable. We will use the notions of model extraction and enumeration as their meaning should be quite intuitive. Related thereto, the term *model expansion* is used in the general first-order case, e.g. in the work of Mitchell and Ternovska [18]. There, an initial (partial) interpretation representing a problem instance is expanded to ultimately find a model of the given theory.

[1] For the other direction, $\Delta^\mathcal{I} = \{a,b \mid A(a) \in \mathcal{A}_\mathcal{I} \text{ or } r(a,b) \in \mathcal{A}_\mathcal{I}\}$, and an individual c not occurring in $\Delta^\mathcal{I}$, we let $c^\mathcal{I} = a$, if $\mathit{Ind}_c(a) \in \mathcal{A}_\mathcal{I}$.

Example 2. We consider the 3-coloring problem for an undirected graph $G = (V, E)$, encoded in $\mathcal{K}_1 = (\mathcal{A}_1, \mathcal{T}_1, \mathcal{R}_1)$, with $\mathcal{T}_1 = \{N \sqsubseteq N_r \sqcup N_g \sqcup N_b,\ N_r \sqsubseteq \forall edge.(N_g \sqcup N_b),\ N_g \sqsubseteq \forall edge.(N_b \sqcup N_r),\ N_b \sqsubseteq \forall edge.(N_r \sqcup N_g), N_r \sqsubseteq \neg N_g, N_r \sqsubseteq \neg N_b, N_g \sqsubseteq \neg N_b\}$. $\mathcal{A}_1 = \{N(v_i) \mid \forall v_i \in V = \{v_1, \ldots, v_n\}\} \cup \{edge(v, v') \mid \forall (v, v') \in E\} \cup \{\neg edge(v, v') \mid \forall (v, v') \in V \times V \setminus E\}$, and $\mathcal{R}_1 = \{\mathsf{Sym}(edge)\}$. Let $\Delta_{\mathcal{K}_1} = \{v_1, \ldots, v_n\}$ be the imposed fixed domain. It is not hard to see, that there is a one-to-one correspondence between the $\Delta_{\mathcal{K}_1}$-models of $\mathcal{K}_1$ and the colorings of G.

Axiomatization of $\Delta_{\mathcal{K}}$-Models When introducing a new semantics for some logic, it is worthwhile to ask if existing reasoners can be used. Indeed, it is easy to see that, assuming $\Delta_{\mathcal{K}} = \{a_1, \ldots, a_n\}$, adding the GCI $\top \sqsubseteq \{a_1, \ldots, a_n\}$ as well as the set of inequality axioms containing $a_i \not\approx a_j$ with $i < j$ to $\mathcal{K}$ will rule out all models of $\mathcal{K}$, not having $\Delta_{\mathcal{K}}$ as their domain. Denoting these additional axioms with $\mathcal{FD}$, we then find that $\mathcal{K}$ is $\Delta_{\mathcal{K}}$-satisfiable iff $\mathcal{K} \cup \mathcal{FD}$ is satisfiable under the classical DL semantics and, likewise, $\mathcal{K} \models_{\text{fd}} \alpha$ iff $\mathcal{K} \cup \mathcal{FD} \models \alpha$ for any axiom α. Consequently, any off-the-shelf $\mathcal{SROIQ}$ reasoner can be used for fixed-domain reasoning, at least when it comes to the classical reasoning tasks.

However, the fact that the currently available DL reasoners are not optimized towards reasoning with axioms of the prescribed type (featuring disjunctions over potentially large sets of individuals) and that available reasoners do not support model extraction and model enumeration led us to develop an alternative computational approach based on ASP.

4 Complexity Analysis

The combined complexity of standard reasoning in $\mathcal{SROIQ}$ is known to be N2ExpTime-complete, both for arbitrary models and finite models [15]. Restricting to fixed domains leads to a drastic drop in complexity. Contrarily, imposing fixed domains on (allegedly) inexpressive fragments such as $\mathsf{DL\text{-}Lite}_{\mathsf{core}}$, turns reasoning into a hard problem.

Therefore, let $\mathsf{DL}_{\mathsf{min}}$ be a minimalistic description logic that merely allows TBox axioms of the form $A \sqsubseteq \neg B$, with $A, B \in \mathsf{N}_C$. Moreover, only atomic assertions of the form $A(a)$ and $r(a, b)$ are admitted. We first demonstrate that satisfiability checking in $\mathsf{DL}_{\mathsf{min}}$ is NP-hard, allowing us to bequeath hardness up to more expressive DLs such as $\mathcal{SROIQ}$. Subsequently, we demonstrate that fixed-domain satisfiability checking in $\mathcal{SROIQ}$ is in NP, and thus obtaining NP-completeness for all languages between $\mathsf{DL}_{\mathsf{min}}$ and $\mathcal{SROIQ}$.

Proposition 1. *The combined complexity of checking fixed-domain satisfiability of a $\mathsf{DL}_{\mathsf{min}}$ knowledge base $\mathcal{K} = (\mathcal{A}, \mathcal{T})$ is* NP*-hard.*

Proof. (Sketch) We obtain hardness by reducing the 3-colorability problem to the following $\mathsf{DL}_{\mathsf{min}}$ axioms. Let $G = (V, E)$ be the input graph. Then, for each node $v_i \in V = \{v_1, \ldots, v_n\}$ we introduce a concept name V_i, and encode the edges as disjointness axioms, such that $\mathcal{T} = \{V_i \sqsubseteq \neg V_j \mid (v_i, v_j) \in E, \forall i, j \in \{1, \ldots, n\}\}$.

The ABox $\mathcal{A}$ consists of the assertions $V_i(a_i)$ for each $V_i \in \{V_1, \ldots, V_n\}$. Now let $\Delta_\mathcal{K} = \{r, g, b\}$, such that under any $\Delta_\mathcal{K}$-fixed interpretation $\mathcal{I}$, necessarily $a_i^\mathcal{I} \in \{r, g, b\}$, $1 \leq i \leq n$. Consequently, G has a 3-coloring, iff $\mathcal{K} = (\mathcal{A}, \mathcal{T})$ is $\Delta_\mathcal{K}$-satisfiable. The reduction is linear in the size of G.

Proposition 2. *The combined complexity of checking fixed-domain satisfiability of $\mathcal{SROIQ}$ knowledge bases is in* NP.

Proof. (Sketch) Let $\mathcal{K}$ be a $\mathcal{SROIQ}$ knowledge base and $\Delta_\mathcal{K}$ be the fixed domain. To show membership, we note that after guessing a $\Delta_\mathcal{K}$-fixed interpretation $\mathcal{I}$, modelhood can be checked in polynomial time. For this we let $\mathcal{C}$ contain all the concept expressions occurring in $\mathcal{K}$ (including subexpressions). Furthermore, let $\mathcal{R}$ contain all role expressions and role chains (including subchains) occurring in $\mathcal{K}$. Obviously, $\mathcal{C}$ and $\mathcal{R}$ are of polynomial size. Then, in a bottom-up fashion, we can compute the extension $C^\mathcal{I}$ of every element C of $\mathcal{C}$ and the extension $r^\mathcal{I}$ of every element r of $\mathcal{R}$ along the defined semantics. Obviously, each such computation step requires only polynomial time. Finally, based on the computed extensions, every axiom of $\mathcal{K}$ can be checked – again in polynomial time.

Combining these propositions yields the following theorem.

Theorem 1. *Fixed-domain satisfiability checking in any language between* DL_min *and $\mathcal{SROIQ}$ is* NP*-complete.*

Note that this finding contrasts with the observation that fixed-domain reasoning in first-order logic is PSPACE-complete. We omit the full proof here, just noting that membership and hardness can be easily shown based on the fact that checking modelhood in FOL is known to be PSPACE-complete [31] and, for the membership part, keeping in mind that NPSPACE = PSPACE thanks to Savitch's Theorem [28]. This emphasizes the fact that, while the fixed-domain restriction turns reasoning in FOL decidable, restricting to $\mathcal{SROIQ}$ still gives a further advantage in terms of complexity (assuming NP $\neq$ PSPACE).

Query Entailment We next consider the complexity of query entailment for DLs. Again, we will notice a very uniform behavior over a wide range of DLs and query types. We will start by showing a hardness result for a very minimalistic setting.

Proposition 3. *The combined complexity of fixed-domain entailment of conjunctive queries from a* DL_min *knowledge base is* Π_2^P*-hard.*

Proof. We show hardness by providing a polynomial reduction from evaluation of quantified Boolean formulae of the form $\Phi = \forall p_1, \ldots, p_\ell \exists q_1, \ldots, q_m \varphi$ such that φ is a Boolean formula where the propositional symbols are from the set $\{p_1, \ldots, p_\ell, q_1, \ldots, q_m\}$. Note that w.l.o.g. we can assume φ to be in conjunctive normal form, i.e. it has the shape $\bigvee L_1 \wedge \ldots \wedge \bigvee L_n$ where the L_i are sets of negated or unnegated propositional symbols.

Given such a formula Φ, we now construct a DL_min knowledge base $\mathcal{K}$, a domain $\Delta_\mathcal{K}$, and a conjunctive query Q (all of polynomial size) such that $\mathcal{K}$ $\Delta_\mathcal{K}$-entails Q if and only if Φ evaluates to true. We let $\Delta_\mathcal{K}$ consist of elements d_t^true and d_t^false for all $t \in \{p_1, \ldots, p_\ell, q_1, \ldots, q_m\}$, and $\mathcal{K}$ consist of the axioms:

- $\mathit{InClause}_L(d_t^{\text{true}})$ whenever $t \in L$ and $\mathit{InClause}_L(d_t^{\text{false}})$ whenever $\neg t \in L$
- $\mathit{compatible}(d_t^{\text{true}}, d_u^{\text{true}})$ and $\mathit{compatible}(d_t^{\text{false}}, d_u^{\text{false}})$ for all $\{t, u\} \subseteq \{p_1, \ldots, p_\ell, q_1, \ldots, q_m\}$
- $\mathit{compatible}(d_t^{\text{false}}, d_u^{\text{true}})$ and $\mathit{compatible}(d_t^{\text{true}}, d_u^{\text{false}})$ for all $\{t, u\} \subseteq \{p_1, \ldots, p_\ell, q_1, \ldots, q_m\}$ with $t \neq u$
- $\mathit{Select}(d_t)$, $C_t(d_t)$ for all $t \in \{p_1, \ldots, p_\ell\}$
- $\mathit{Select}(d_t^{\text{true}})$ and $\mathit{Select}(d_t^{\text{false}})$ for all $t \in \{q_1, \ldots, q_m\}$
- $C_t(d_t^{\text{true}})$, $C_t(d_t^{\text{false}})$ for all $t \in \{p_1, \ldots, p_\ell, q_1, \ldots, q_m\}$
- $C_t \sqcap C_u \sqsubseteq \bot$ for all $\{t, u\} \in \{p_1, \ldots, p_\ell, q_1, \ldots, q_m\}$ with $t \neq u$

Finally, we let Q be the conjunctive query using the variables $x_{L_1}, \ldots, x_{L_n}$ and consisting of the atoms $\mathit{InClause}_L(x_L)$, $\mathit{Select}(x_L)$ for all $L \in \{L_1, \ldots, L_n\}$ as well as $\mathit{compatible}(x_L, x_{L'})$ for all $\{L, L'\} \in \{L_1, \ldots, L_n\}$.

We now sketch the argument why the above claimed correspondence holds. By construction, the minimal $\Delta_{\mathcal{K}}$-models $\mathcal{I}$ for every $i \in \{1, \ldots, m\}$ are exactly those where (next to the explicitly stated concept and role memberships) either $d_{p_i}^{\text{true}} \in \mathit{Select}^{\mathcal{I}}$ or $d_{p_i}^{\text{false}} \in \mathit{Select}^{\mathcal{I}}$ holds. Consequently Q is entailed, iff for each of these models (representing all possible truth assignments to $p_1, \ldots, p_\ell$), one literal from every clause L_i can be selected such that (a) this selection is consistent (i.e., no contradicting literals are selected) and (b) whenever a literal w.r.t. $p_1, \ldots, p_\ell$ is selected, it must be the one corresponding with the model's predefined truth assignment for these propositional symbols. However, this is the case exactly if Φ is valid.

We continue by showing that even for very expressive DLs and query languages, query entailment under the fixed domain semantics is still in the second level of the polynomial hierarchy.

Proposition 4. *The combined complexity of the fixed-domain entailment of bounded-arity Datalog queries from a $\mathcal{SROIQ}$ knowledge base is in Π_2^{P}.*

Proof. Satisfaction of a bounded-arity Datalog query in a database (or finite interpretation) is in NP: there are only polynomially many ground atoms that can be derived, hence, whenever the query is entailed, there is a ground proof tree of polynomial size which can be verified in polynomial time. Consequently, fixed-domain non-entailment of such a query Q from a $\mathcal{SROIQ}$ knowledge base $\mathcal{K}$ can be realized by (a) guessing an interpretation $\mathcal{I}$ (b) verifying $\mathcal{I} \models \mathcal{K}$ in polynomial time (cf. the proof of Proposition 2) and (c) using an NP oracle to verify $\mathcal{I} \not\models Q$. Consequently, checking fixed-domain entailment is in $\text{coNP}^{\text{NP}} = \Pi_2^{\text{P}}$.

Bounded-arity Datalog queries over DLs are rather expressive, they subsume many of the prominent query classes in knowledge representation and databases, including (unions of) conjunctive queries, positive queries, (unions of) conjunctive 2-way regular path queries [7], positive 2-way regular path queries, (unions of) conjunctive nested 2-way regular path queries [3] and regular queries as defined in [24]. Combining the two propositions, we obtain the following theorem.

Theorem 2. *For any class of queries subsuming conjunctive queries and subsumed by bounded-arity Datalog queries and any DL subsuming* $\mathsf{DL}_{\mathsf{min}}$ *and subsumed by* $\mathcal{SROIQ}$*, the combined complexity of fixed-domain query entailment is* Π_2^{P}*-complete.*

5 Practical Fixed-Domain Reasoning

In Section 3 we already claimed that available reasoners perform poorly on knowledge bases when axiomatizing the fixed-domain semantics, and we support this statement with an evaluation in the sequel (cf. Section 5.3). Thus, a more viable approach is required when considering practical reasoning. To this end, we propose an encoding of arbitrary $\mathcal{SROIQ}$ knowledge bases into *answer set programs*. This allows us to use existing ASP machinery to perform both standard reasoning as well as the non-standard tasks *model extraction & enumeration* and query entailment quite elegantly.

We review the basic notions of answer set programming [22] under the stable model semantics [10], for further details we refer to [4]. We consider atoms, predicates and terms as defined in Section 2. Each term is either a variable or a constant from a domain $\mathcal{U}$. An atom is *ground* if it is free of variables. $B_{\mathcal{U}}$ denotes the set of all ground atoms over $\mathcal{U}$. A *(disjunctive) rule* ρ is of the form

$$a_1 \vee \cdots \vee a_n \leftarrow b_1, \ldots, b_k, \; not\, b_{k+1}, \ldots, \; not\, b_m,$$

with $n \geq 0$, $m \geq k \geq 0$, $n + m > 0$, where $a_1, \ldots, a_n$, $b_1, \ldots, b_m$ are atoms, and "*not*" stands for *default negation*. The *head* of ρ is the set $H(\rho) = \{a_1, \ldots, a_n\}$ and the *body* of ρ is $B(\rho) = \{b_1, \ldots, b_k, \; not\, b_{k+1}, \ldots, \; not\, b_m\}$. Furthermore, $B^+(\rho) = \{b_1, \ldots, b_k\}$ and $B^-(\rho) = \{b_{k+1}, \ldots, b_m\}$. An *interpretation* $I \subseteq B_{\mathcal{U}}$ *satisfies* a ground rule ρ iff $H(\rho) \cap I \neq \emptyset$ whenever $B^+(\rho) \subseteq I$, $B^-(\rho) \cap I = \emptyset$. I satisfies a ground program Π, if each $\rho \in \Pi$ is satisfied by I. A non-ground rule ρ (resp., a program Π) is satisfied by an interpretation I iff I satisfies all groundings of ρ (resp., $Gr(\Pi)$). $I \subseteq B_{\mathcal{U}}$ is an *answer-set* of Π iff it is a subset-minimal set satisfying the reduct $\Pi^I = \{H(\rho) \leftarrow B^+(\rho) \mid I \cap B^-(\rho) = \emptyset, \rho \in Gr(\Pi)\}$. For a program Π, we denote the set of its answer-sets by $\mathcal{AS}(\Pi)$.

5.1 ASP Encodings of DL Knowledge Bases

Due to the identified complexity results, our ASP based approach suits perfectly for all involved reasoning tasks [4]. Intuitively, the set of all $\Delta_{\mathcal{K}}$-interpretations defines a search space, which can be traversed searching for $\Delta_{\mathcal{K}}$-models, guided by appropriate constraints. We thus propose a translation $\Pi(\mathcal{K})$ for any $\mathcal{SROIQ}$ knowledge base $\mathcal{K}$; i.e. $\Pi(\mathcal{K}) = \Pi_{\mathrm{gen}}(\mathcal{K}) \cup \Pi_{\mathrm{chk}}(\mathcal{K})$, consisting of a generating part $\Pi_{\mathrm{gen}}(\mathcal{K})$ that defines all potential candidate interpretations, and a constraining part $\Pi_{\mathrm{chk}}(\mathcal{K})$ that rules out interpretations violating axioms in $\mathcal{K}$. However, we can only sketch the translation and refer to [8], where we introduced the main idea of the translation already.

The knowledge base is required to be in normalized form, obtained by a modified structural transformation $\Omega(\mathcal{K})$, based on the one proposed in [21]. 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, $\{a\}$, $\forall r.B$, $\exists r.\mathit{Self}$, $\neg\exists r.\mathit{Self}$, $\geq n\, r.B$, or $\leq n\, r.B$, for B a literal concept, r a role, and n a positive integer. A normalized knowledge base $\Omega(\mathcal{K})$ is a model-conservative extension of $\mathcal{K}$, i.e. every ($\Delta_{\mathcal{K}}$-) model of $\Omega(\mathcal{K})$ is a ($\Delta_{\mathcal{K}}$-) model of $\mathcal{K}$ and every ($\Delta_{\mathcal{K}}$-) model of $\mathcal{K}$ can be turned into a ($\Delta_{\mathcal{K}}$-) model of $\Omega(\mathcal{K})$ by finding appropriate interpretations for the concepts and roles introduced by Ω. Thereby it is straightforward to extract a model for $\mathcal{K}$, given a model of $\Omega(\mathcal{K})$.

Candidate Generation Following the generate & test paradigm, we let $\Pi_{\text{gen}}(\mathcal{K})$ be the program that generates (all) possible interpretations over $\Delta_{\mathcal{K}}$; i.e. for each concept name A, role name r, and individual a all possible extensions over $\Delta_{\mathcal{K}}$ are generated. Thus, an answer-set $\mathbf{A}$ of $\Pi_{\text{gen}}(\mathcal{K})$ directly induces an interpretation $\mathcal{I}_{\mathbf{A}}$ of $\mathcal{K}$ over the fixed-domain $\Delta_{\mathcal{K}}$. We denote the set of all interpretations of $\mathcal{K}$ over $\Delta_{\mathcal{K}}$ with $\mathcal{B}_{\mathcal{K}}$.

Proposition 5. *Let $\mathcal{K}$ be a $\mathcal{SROIQ}$ knowledge base and $\Pi_{\text{gen}}(\mathcal{K})$ the obtained logic program. Then, it holds that $\mathcal{B}_{\mathcal{K}}$ coincides with the set of all answer-sets of $\Pi_{\text{gen}}(\mathcal{K})$.*

Axiom Encoding For the test part, we turn each axiom $\alpha \in \mathcal{T} \cup \mathcal{R}$ into a constraint, ultimately ruling out those candidate interpretations not satisfying α, whence $\Pi_{\text{chk}}(\mathcal{K}) = \Pi_{\text{chk}}(\mathcal{T}) \cup \Pi_{\text{chk}}(\mathcal{R})$. Since each $\alpha \in \mathcal{T}$ is of the form $\top \sqsubseteq \bigsqcup_{i=1}^{n} C_i$, we simply turn it into a negative constraint of the form $\bigsqcap_{i=1}^{n} \neg C_i \sqsubseteq \bot$, and add its direct translation to $\Pi_{\text{chk}}(\mathcal{T})$. Role assertions and role inclusion axioms are also turned into constraints, and we add their direct translation to $\Pi_{\text{chk}}(\mathcal{R})$.

Theorem 3. *For any normalized $\mathcal{SROIQ}$ knowledge base $\mathcal{K} = (\mathcal{A}, \mathcal{T}, \mathcal{R})$ and its translation $\Pi(\mathcal{K})$, it holds $\mathcal{AS}(\Pi(\mathcal{K})) = \{\mathbf{B} \mid \mathbf{B} \in \mathcal{B}_{\mathcal{K}}$ and $\mathcal{I}_{\mathbf{B}} \models \mathcal{K}\}$.*

With this theorem in place, we benefit from the translation in many aspects. Most notably, in addition to the standard DL reasoning tasks, *model extraction* and *model enumeration* can be carried out without additional efforts, since both are natural tasks for answer set solvers. Moreover, all mentioned query formalisms can be straightforwardly expressed in a rule-based way, whence integration in our framework is immediate.

Example 3. We reconsider $\mathcal{K}_1$ from Example 2. The axioms (a) $N \sqsubseteq N_r \sqcup N_g \sqcup N_b$ and (b) $N_r \sqsubseteq \forall \mathit{edge}.(N_g \sqcup N_b)$, yield the following constraints:

$$\leftarrow N(X), \mathit{not}\, N_r(X), \mathit{not}\, N_g(X), \mathit{not}\, N_b(X). \tag{1}$$

$$\leftarrow N_r(X), \mathit{edge}(X,Y), \mathit{not}\, A_{N_g \sqcup N_b}(Y). \tag{2}$$

$$\leftarrow A_{N_g \sqcup N_b}(X), \mathit{not}\, N_g(X), \mathit{not}\, N_b(X). \tag{3}$$

Due to normalization, (b) results in constraints (2) & (3), ensuring that if there is an edge from some red node X to Y, necessarily Y is either green or blue.

5.2 Prototype Implementation

We implemented our translation based approach as an open-source tool – named `Wolpertinger`.[2] The obtained logic programs can be evaluated with most modern ASP solvers. However, the evaluation was conducted using `Clingo` [9] for grounding and solving, since it currently is the most prominent solver leading the latest competitions [5]. We present preliminary evaluation results based on simple ontologies, encoding constraint-satisfaction-type combinatorial problems. Existing OWL ontologies typically used for benchmarking, e.g. SNOMED or GALEN [29, 23], do not fit our purpose, since they are modeled with the classical semantics in mind and often have little or no ABox information.

Our tests provide runtimes compared to the popular `HermiT` reasoner [11] and `Konclude`[30]. Whereas a direct comparison would not be fair, the conducted tests shall merely show the feasibility of our approach in comparison to standard DL reasoners using the axiomatization. In particular we focus on model enumeration, for which we can not conduct any comparison with existing DL reasoners. The evaluation itself is conducted on a standard desktop machine (Unix operating system, 2.7 Ghz Intel Core i5 Processor, 8 GB memory and standard Java-VM settings).

5.3 Initial Experiments

Unsatisfiability We construct an unsatisfiable knowledge base $\mathcal{K}_n = (\mathcal{A}_n, \mathcal{T}_n, \emptyset)$, with $\mathcal{T}_n = \{A_1 \sqsubseteq \exists r.A_2, \ldots, A_n \sqsubseteq \exists r.A_{n+1}, A_i \sqcap A_j \sqsubseteq \bot \mid 1 \leq i < j \leq n+1\}$ and $\mathcal{A}_n = \{A_1(a_1), \top(a_1), \ldots, \top(a_n)\}$, together with the fixed-domain $\Delta_\mathcal{K} = \{a_1, \ldots, a_n\}$. Inspired by common pigeonhole-type problems, we have $\mathcal{K}_n$ enforce an r-chain of length $n+1$ without repeating elements, yet, having fixed $\Delta_\mathcal{K}$ to n elements such a model cannot exist. Table 1 depicts the runtimes for detecting unsatisfiability of $\mathcal{K}_n$, for increasing n. The durations correspond to the pure solving time as stated by the tools (including grounding in the case of `Clingo`), and neglecting pre-processing time. As the figures suggest, $\mathcal{K}_n$ is a potential worst-case scenario, where any of the tools is doomed to test all combinations. Whereas `Wolpertinger` is faster in claiming inconsistency in all cases up to $\mathcal{K}_{10}$, `HermiT` is slightly faster up from $\mathcal{K}_{11}$ – both leaving `Konclude` behind. However, $\mathcal{K}_{12}$ is already beyond a feasible time bound for all reasoners.

Model Extraction and Enumeration With Table 2, we next provide some figures for model extraction and partial enumeration (retrieving a given number of $\Delta_\mathcal{K}$ models). To this end, we created a knowledge base modeling fully and correctly filled Sudokus, featuring 108 named individuals, 13 concept names and 1 role name. When invoking a satisfiability test on this knowledge base (axiomatized) using `HermiT` & `Konclude`, no answer was given within 15 minutes. On average, a solution for a given Sudoku instance is provided in around 7 seconds, of which more than 6 seconds are needed for grounding, while the actual solving is done in

[2] `https://github.com/wolpertinger-reasoner/Wolpertinger`

Table 1. Runtimes: Detecting unsatisfiability of $\mathcal{K}_n$.

#	$\mathcal{K}_n$	`Wolpertinger`	`HermiT`	`Konclude`
1	5	< 0.01 s	0.48 s	0.04 s
2	6	< 0.01 s	0.67 s	0.07 s
3	7	0.04 s	0.94 s	0.26 s
4	8	0.33 s	1.81 s	1.79 s
5	9	3.72 s	9.52 s	16.19 s
6	10	68.53 s	87.88 s	152.37 s
7	11	1 095.49 s	1 027.33 s	1 682.41 s

Table 2. Runtimes: Enumerating Sudoku Instances.

#	**Models**	**Time(Total)**	**Time(Solving)**
1	100	6.73 s	0.11 s
2	1 000	7.16 s	0.33 s
3	10 000	9.06 s	2.39 s
4	100 000	29.27 s	22.53 s
5	1 000 000	225.40 s	218.56 s

less than 0.1 seconds. For model enumeration, we used the knowledge base but removed information concerning pre-filled cells, turning the task into generating new Sudoku instances. The size of the grounded program is 20 MB, which takes around 6 seconds to obtain, as reflected in Table 2.

6 Conclusion

For OWL ontologies which represent constraint-type problems, the fixed-domain semantics allows to confine modelhood of interpretations towards more intuitional models. Although modeling features are limited, we argue that quite large and involved problem scenarios can be modeled in OWL ontologies. Clearly, evaluations of our system with respect to such ontologies remain as imperative issue. Moreover, we will consider translations into other formalisms, such as pure CSP languages, or even SAT, as future work. While remaining in monotonic waters, prospective considerations are in the direction of non-monotonic semantics. As such, rule-based extensions of OWL – monotonic [14, 20] or nonmonotonic [19, 1] – should be straightforward to accommodate. Moreover, we plan to incorporate typical ontology engineering tasks such as explanation and axiom pinpointing into our ASP-based framework.

Acknowledgements

We are grateful for all the valuable feedback from our colleagues and the anonymous workshop reviewers, which helped greatly to improve this work.

References

1. Alferes, J.J., Knorr, M., Swift, T.: Query-Driven Procedures for Hybrid MKNF Knowledge Bases. ACM Transactions on Computational Logic 14(2), 16:1–16:43 (2013)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, second edn. (2007)
3. Bienvenu, M., Calvanese, D., Ortiz, M., Simkus, M.: Nested regular path queries in description logics. In: Baral, C., Giacomo, G.D., Eiter, T. (eds.) Proc. of the 14th Conference on Principles of Knowledge Representation and Reasoning (KR 2014). AAAI Press (2014)
4. Brewka, G., Eiter, T., Truszczyński, M.: Answer set programming at a glance. Commun. ACM 54(12), 92–103 (2011)
5. Calimeri, F., Gebser, M., Maratea, M., Ricca, F.: Design and results of the 5th answer set programming competition. Artif. Intell. 231, 151–181 (2016)
6. Calvanese, D.: Finite model reasoning in description logics. In: Padgham, L., Franconi, E., Gehrke, M., McGuinness, D.L., Patel-Schneider, P.F. (eds.) Proc. of the International Workshop on Description Logics (DL 1996). AAAI Technical Report, vol. WS-96-05, pp. 25–36. AAAI Press (1996)
7. Calvanese, D., Eiter, T., Ortiz, M.: Regular path queries in expressive description logics with nominals. In: Boutilier, C. (ed.) Proc. of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009). pp. 714–720 (2009)
8. Gaggl, S.A., Rudolph, S., Schweizer, L.: Bound Your Models! How to Make OWL an ASP Modeling Language. In: Ellmauthaler, S., Schulz, C. (eds.) Proc. of the International Workshop on User-Oriented Logic Programming (IULP 2015).
9. Gebser, M., Roland Kaminski, B.K., Schaub, M.O.T., Schneider, M.T.: Potassco: The Potsdam Answer Set Solving Collection. AI Communications 24(2), 107–124 (2011)
10. Gelfond, M., Lifschitz, V.: Classical negation in logic programs and disjunctive databases. New Generation Comput. 9(3/4), 365–386 (1991)
11. Glimm, B., Horrocks, I., Motik, B., Stoilos, G., Wang, Z.: HermiT: an OWL 2 reasoner. Journal of Automated Reasoning 53(3), 245–269 (2014)
12. Horridge, M., Patel-Schneider, P.F.: OWL 2 Web Ontology Language Manchester Syntax (2012)
13. Horrocks, I., Kutz, O., Sattler, U.: The Even More Irresistible $\mathcal{SROIQ}$. In: Doherty, P., Mylopoulos, J., Welty, C.A. (eds.) Proc. of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006). pp. 57–67. AAAI Press (2006)
14. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B.N., Dean, M.: SWRL: A Semantic Web Rule Language. W3C Member Submission (21 May 2004)
15. Kazakov, Y.: $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are harder than $\mathcal{SHOIQ}$. In: Brewka, G., Lang, J. (eds.) Proc. of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR 2008). pp. 274–284. AAAI Press (2008)
16. Knublauch, H., Musen, M.A., Rector, A.L.: Editing Description Logic Ontologies with the Protégé OWL Plugin. In: Haarslev, V., Möller, R. (eds.) Proc. of the International Workshop on Description Logics (DL 2004). CEUR Workshop Proceedings, vol. 104. CEUR-WS.org (2004)
17. Lutz, C., Sattler, U., Tendera, L.: The complexity of finite model reasoning in description logics. Information and Computation 199(1-2), 132–171 (May 2005)

18. Mitchell, D.G., Ternovska, E.: A framework for representing and solving NP search problems. In: Veloso, M.M., Kambhampati, S. (eds.) Proc. of the 20th National Conference on Artificial Intelligence and the 17th Innovative Applications of Artificial Intelligence Conference (AAAI 2005). pp. 430–435. AAAI Press / The MIT Press (2005)
19. Motik, B., Rosati, R.: Reconciling description logics and rules. Journal of the ACM 57(5) (2010)
20. Motik, B., Sattler, U., Studer, R.: Query Answering for OWL DL with Rules. Journal of Web Semantics 3(1), 41–60 (2005)
21. Motik, B., Shearer, R., Horrocks, I.: Hypertableau Reasoning for Description Logics. Artificial Intelligence Research 36, 165–228 (2009)
22. Niemelä, I.: Logic programs with stable model semantics as a constraint programming paradigm. Ann. Math. Artif. Intell. 25(3-4), 241–273 (1999)
23. Rector, A., Horrocks, I.: Experience building a large, re-usable medical ontology using a description logic with transitivity and concept inclusions. In: Proc. of the Workshop on Ontological Engineering (1997)
24. Reutter, J.L., Romero, M., Vardi, M.Y.: Regular Queries on Graph Databases. In: Arenas, M., Ugarte, M. (eds.) Proc. of the 18th International Conference on Database Theory (ICDT 2015). LIPIcs, vol. 31, pp. 177–194. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik (2015)
25. Rosati, R.: Finite Model Reasoning in DL-Lite. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) Proc. of the 5th European Semantic Web Conference (ESWC 2008). LNCS, vol. 5021, p. 215. Springer (2008)
26. Rudolph, S.: Foundations of Description Logics. In: Polleres, A., d'Amato, C., Arenas, M., Handschuh, S., Kroner, P., Ossowski, S., Patel-Schneider, P.F. (eds.) Reasoning Web. 7th International Summer School 2011, Tutorial Lectures. LNCS, vol. 6848, pp. 76–136. Springer (2011)
27. Rudolph, S.: Undecidability Results for Database-Inspired Reasoning Problems in Very Expressive Description Logics. In: Baral, C., Delgrande, J., Wolter, F. (eds.) Proc. of the 15th International Conference on Principles of Knowledge Representation and Reasoning (KR 2016), AAAI Press (2016).
28. Savitch, W.J.: Relationships Between Nondeterministic and Deterministic Tape Complexities. J. Comput. Syst. Sci. 4(2), 177–192 (Apr 1970)
29. Spackman, K.A., Campbell, K.E., Côté, R.A.: SNOMED RT: a reference terminology for health care. In: AMIA 1997, American Medical Informatics Association Annual Symposium. AMIA (1997)
30. Steigmiller, A., Liebig, T., Glimm, B.: Konclude: System description. Journal of Web Semantics 27, 78–85 (2014)
31. Stockmeyer, L.J.: The Complexity of Decision Problems in Automata Theory and Logic. Ph.D. thesis, Massachusetts Institute of Technology (1974)
32. W3C OWL Working Group: OWL 2 Web Ontology Language: Document Overview. W3C Recommendation (2009)