Worst-case Optimal Reasoning for the Horn-DL Fragments of OWL 1 and 2*

Magdalena Ortiz† and **Sebastian Rudolph**‡ and **Mantas Šimkus**†

†Vienna University of Technology, Austria
{ortiz,simkus}@kr.tuwien.ac.at

‡Karlsruhe Institute of Technology, Germany
rudolph@kit.edu

Abstract

Horn fragments of Description Logics (DLs) have gained popularity because they provide a beneficial trade-off between expressive power and computational complexity and, more specifically, are usually tractable w.r.t. data complexity. Despite their potential, and partly due to the intricate interaction of nominals ($\mathcal{O}$), inverses ($\mathcal{I}$) and counting ($\mathcal{Q}$), such fragments had not been studied so far for the DLs $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$ that underly OWL 1 and 2. In this paper, we present a polynomial and modular translation from Horn-$\mathcal{SHOIQ}$ knowledge bases into DATALOG, which shows that standard reasoning tasks are feasible in deterministic single exponential time. This improves over the previously known upper bounds, and contrasts the known NEXPTIME completeness of full $\mathcal{SHOIQ}$. Thereby, Horn-$\mathcal{SHOIQ}$ stands out as the first EXPTIME complete DL that allows simultaneously for $\mathcal{O}$, $\mathcal{I}$, and $\mathcal{Q}$. In addition, we show that standard reasoning in Horn-$\mathcal{SROIQ}$ is 2-EXPTIME complete. Despite their high expressiveness, both Horn-$\mathcal{SHOIQ}$ and Horn-$\mathcal{SROIQ}$ have polynomial data complexity. This makes them particularly attractive for reasoning in semantically enriched systems with large data sets. A promising first step in this direction could be achieved exploiting existing DATALOG engines, along the lines of our translation.

Introduction

In the Semantic Web, an *ontology* specifies a common conceptualization of an application domain. This conceptualization should support access to distributed data repositories by different applications, and enable software agents to exchange information independently of their possibly different internal representations. Such a view of ontologies as means to access complex data sources is also advocated in other areas like ontology-based data access, data and information integration and peer-to-peer data management. As specification formalisms for these ontologies, the Web Ontology Languages (OWL) proposed by the World Wide Web Consortium (W3C) are now a widely accepted standard.

The OWL languages are based on *Description Logics* (DLs), a family of expressive languages for knowledge representation (Baader *et al.* 2007). In particular, OWL 1 DL and OWL 2 DL, the most expressive decidable species of the OWL standards, are based on the DLs known as $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$. The distinguishing feature of these two DLs is the presence of nominals ($\mathcal{O}$), inverses ($\mathcal{I}$) and counting ($\mathcal{Q}$). The combination is considered hard to handle, and it is known to increase the complexity of reasoning w.r.t. other related DLs (see later). $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$ are receiving wide attention, and several authors have presented algorithms to support automated reasoning over their ontologies in the last years (cf. (Horrocks & Sattler 2005; Kazakov & Motik 2008; Tobies 2000)).

The *data complexity of reasoning* plays an increasingly important role in DL applications. In particular, when DL ontologies are used as formalizations of complex data sources, it is imperative that efficiency of reasoning scales well in the presence of large amounts of data. Formally, the data complexity of a DL is the complexity of reasoning in knowledge bases (KBs) where the terminological part is assumed to be fixed while the assertional information (ABox) varies. The intractability of data complexity in most standard DLs, including very restricted sublogics of $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$, fostered the development of tailored DLs, where syntactic restrictions are applied to reduce the complexity, e.g., the DL-*Lite* family (Calvanese *et al.* 2007), and $\mathcal{EL}$ (Baader 2003). Among other restrictions, these logics prohibit disjunction, and they can be seen as *Horn fragments* of first order logic. In (Hustadt, Motik, & Sattler 2005), the Horn fragment of $\mathcal{SHIQ}$ (the DL underlying OWL 1 Lite) was introduced and its polynomial data complexity established (as opposed to CO-NP completeness in full $\mathcal{SHIQ}$). Since then, Horn fragments of other DLs have gained attention (cf. (Krötzsch, Rudolph, & Hitzler 2007)).

In this paper we study the Horn fragments of $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$, and provide a DATALOG-based reasoning method that yields optimal upper bounds for the combined and data complexity of the two fragments. Our main contributions can be summarized as follows:

- We present a polynomial and modular reduction from KB satisfiability in Horn-$\mathcal{SHOIQ}$ to consistency of a DATALOG program. This novel reasoning method shows that the former problem can be decided in deterministic single exponential time in the size of the input knowledge base. The result extends to other *standard reasoning tasks* over Horn-$\mathcal{SHOIQ}$ knowledge bases such as subsumption and instance checking, which are polynomially reducible to KB satisfiability (Krötzsch, Rudolph, & Hitzler 2007).
- The EXPTIME upper bound we obtain for standard reasoning in Horn-$\mathcal{SHOIQ}$ is tight. It improves over the

*This work was partially supported by (†) the Austrian Science Fund (FWF) grant P20840, the EC project OntoRule (IST-2009-231875), the (CONACYT) grant 187697, and (‡) the project ExpresST funded by the Deutsche Forschungsgemeinschaft (DFG).
Copyright © 2010, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

previously known upper bounds, and contrasts the known NEXPTIME completeness of full $\mathcal{SHOIQ}$. Notably, Horn-$\mathcal{SHOIQ}$ stands out as the first DL in EXPTIME complete DL allowing simultaneously for $\mathcal{O}$, $\mathcal{I}$, and $\mathcal{Q}$.

- We show that standard reasoning in Horn-$\mathcal{SROIQ}$ is 2-EXPTIME complete. The upper bound follows from our DATALOG reduction together with a (possibly exponential) translation from Horn-$\mathcal{SROIQ}$ into Horn-$\mathcal{SHOIQ}$. The lower bound is given by a reduction of a deterministic Turing machine running in double exponential time. In contrast, full $\mathcal{SROIQ}$ is 2-NEXPTIME complete (Kazakov 2008).
- Finally, via the DATALOG encoding, we show that despite their high expressiveness, both Horn-$\mathcal{SHOIQ}$ and Horn-$\mathcal{SROIQ}$ have polynomial data complexity.

In summary, our results identify Horn-$\mathcal{SHOIQ}$ and Horn-$\mathcal{SROIQ}$ as particularly attractive DLs for knowledge representation and reasoning in semantically enriched systems with large data sets. A promising next step in this direction could be to exploit existing efficient DATALOG engines for reasoning over ontologies in these expressive logics, along the lines of our translation.

The DLs Horn-$\mathcal{SROIQ}$ and Horn-$\mathcal{SHOIQ}$

The DL $\mathcal{SROIQ}$ was introduced in (Horrocks, Kutz, & Sattler 2006). Here we define Horn-$\mathcal{SROIQ}$, a fragment of $\mathcal{SROIQ}$ that disallows disjunction, establishing a correspondence to a Horn fragment of first-order logic with equality. Without loss of generality, we focus on a normal form close to the one in (Krötzsch, Rudolph, & Hitzler 2007).

As usual, we assume countably infinite sets $\mathsf{N_C}$, $\mathsf{N_R}$ and $\mathsf{N_I}$ of *concept names*, *role names*, and *individuals* respectively; we also assume $\{\top, \bot\} \subset \mathsf{N_C}$. If $p \in \mathsf{N_R}$, then p and p^- are *roles*, and their respective *inverses* are $\bar{p} = p^-$ and $\overline{p^-} = p$.

As in full $\mathcal{SROIQ}$, a *generalized role inclusion axiom* (RIA) has the form $s_1 \circ \ldots \circ s_n \sqsubseteq r$, where r and each s_i are roles. A set of RIAs $\mathcal{R}$ is *regular* if there is a strict partial order $\prec$ on the roles such that $s \prec r$ iff $\bar{s} \prec r$, and each RIA in $\mathcal{R}$ is of one of the forms (i) to (v) in Table 1. The *simple roles* in $\mathcal{R}$ are defined inductively: (a) $p \in \mathsf{N_R}$ is simple if each RIA with p in its right-hand-side has the form $s \sqsubseteq p$ for some simple s; and (b) p^- is simple if p is simple.

A *TBox* $\mathcal{T}$ is of the form $\mathcal{R} \cup \mathcal{T}'$, where $\mathcal{R}$ is a regular set of RIAs and $\mathcal{T}'$ is a set of axioms of the forms (vi) to (xii) in Table 1.[1] A Horn-$\mathcal{SROIQ}$ knowledge base (KB) is a tuple $\mathcal{K}=\langle \mathcal{T}, \mathcal{A}\rangle$, where $\mathcal{T}$ is a TBox and the *ABox* $\mathcal{A}$ is a non-empty set of *assertions* of the forms (xiii) and (xiv).

The DL Horn-$\mathcal{SHOIQ}$ is obtained by restricting Horn-$\mathcal{SROIQ}$ as follows: 1. only RIAs of the forms (i), (ii') and (v) are allowed; 2. *disjointness axioms* (vi) are disallowed; and 3. concepts of the form $\exists r.\mathrm{Self}$ are disallowed.

The semantics of KBs is given by *interpretations* $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$, which map each $a \in \mathsf{N_I}$ to some $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, each $A \in \mathsf{N_C}$ to some $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and each $r \in \mathsf{N_R}$ to some $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, such that $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$. The map $\cdot^{\mathcal{I}}$ is extended to all concepts and roles as usual, cf. Table 1. We say that $\mathcal{I}$ is a model of $\mathcal{K}$, in symbols $\mathcal{I} \models \mathcal{K}$, if it satisfies all axioms in the TBox and all assertions in the ABox. The *KB satisfiability* problem consists on deciding, for a given a KB $\mathcal{K}$, whether there is an $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$.

	Syntax	Semantics	
(i)	$p^- \sqsubseteq p$	$p^{-\mathcal{I}} \subseteq p^{\mathcal{I}}$	TBox
(ii)	$w \sqsubseteq p$	$w^{\mathcal{I}} \subseteq p^{\mathcal{I}}$	
(ii')	$s_1 \sqsubseteq p$	${s_1}^{\mathcal{I}} \subseteq p^{\mathcal{I}}$	
(iii)	$p \circ w \sqsubseteq p$	$p^{\mathcal{I}} \circ w^{\mathcal{I}} \subseteq p^{\mathcal{I}}$	
(iv)	$w \circ p \sqsubseteq p$	$w^{\mathcal{I}} \circ p^{\mathcal{I}} \subseteq p^{\mathcal{I}}$	
(v)	$p \circ p \sqsubseteq p$	$p^{\mathcal{I}} \circ p^{\mathcal{I}} \subseteq p^{\mathcal{I}}$	
(vi)	$\mathsf{Disj}(s, s')$	$s^{\mathcal{I}} \cap {s'}^{\mathcal{I}} = \emptyset$	
(vii)	$A \sqcap B \sqsubseteq C$	$A^{\mathcal{I}} \cap B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$	
(viii)	$A \sqsubseteq \forall r.B$	$A^{\mathcal{I}} \subseteq \{d \mid \forall d'.\langle d, d'\rangle \in r^{\mathcal{I}} \rightarrow d' \in B^{\mathcal{I}}\}$	
(ix)	$A \sqsubseteq \exists r.B$	$A^{\mathcal{I}} \subseteq \{d \mid \exists d'.\langle d, d'\rangle \in r^{\mathcal{I}} \wedge d' \in B^{\mathcal{I}}\}$	
(x)	$\exists r.A \sqsubseteq B$	$\{d \mid \exists d'.\langle d, d'\rangle \in r^{\mathcal{I}} \wedge d' \in B^{\mathcal{I}}\} \subseteq A^{\mathcal{I}}$	
(xi)	$A \sqsubseteq \leqslant 1\, s.B$	$A^{\mathcal{I}} \subseteq \{d \mid N^{\mathcal{I}}_{s,B}(d) \leq 1\}$	
(xii)	$A \sqsubseteq \geqslant m\, s.B$	$A^{\mathcal{I}} \subseteq \{d \mid N^{\mathcal{I}}_{s,B}(d) \geq m\}$	
(xiii)	$a{:}A$	$a^{\mathcal{I}} \in A^{\mathcal{I}}$	ABox
(xiv)	$(a, b){:}r$	$\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in r^{\mathcal{I}}$	

Table 1: Syntax and semantics of Horn-$\mathcal{SROIQ}$. Here $m \geq 0$, $a, b \in \mathsf{N_I}$, $p \in \mathsf{N_R}$, r is a role, and s and s' are simple roles. Further, $w = s_1 \circ \ldots \circ s_n$ $n \geq 1$ and, for each $1 \leq i \leq n$, s_i is a role with $s_i \prec p$. *Concepts* A, B, and C are either concept names, *nominals* of the form $\{a\}$, or of the form $\exists p.\mathrm{Self}$. For an interpretation $\mathcal{I}$, we have $w^{\mathcal{I}} = {s_1}^{\mathcal{I}} \circ \ldots \circ {s_n}^{\mathcal{I}}$, where $\circ$ is overridden to denote relational composition, $(p^-)^{\mathcal{I}} = \{\langle d', d\rangle \mid \langle d, d'\rangle \in p^{\mathcal{I}}\}$, $\{a\}^{\mathcal{I}} = \{a^{\mathcal{I}}\}$, and $(\exists p.\mathrm{Self})^{\mathcal{I}} = \{d \mid \langle d, d\rangle \in p^{\mathcal{I}}\}$. Finally, $N^{\mathcal{I}}_{s,B}(d)$ denotes the cardinality of the set $\{d' \mid \langle d, d'\rangle \in s^{\mathcal{I}} \wedge d' \in B^{\mathcal{I}}\}$. The boldface roman numbers indicate the expressions that may appear in normal Horn-$\mathcal{ALCHOIQ}^{\mathsf{Disj}}_{\mathsf{Self}}$ KBs.

Finally, we observe that every Horn-$\mathcal{SROIQ}$ KB $\mathcal{K}$ can be rewritten into an equisatisfiable $\mathcal{K}'$ as follows: 1. axioms of the form $\exists r.A \sqsubseteq B$ are rewritten as $A \sqsubseteq \forall \bar{r}.B$; 2. $\geqslant m\, s.B$ is expressed via auxiliary roles, Disj and $\exists$ (cf. (Rudolph, Krötzsch, & Hitzler 2008)); and 3. RIAs with non-simple roles are eliminated using the standard automaton encoding provided in (Kazakov 2008). The resulting $\mathcal{K}'$ is a *normal Horn-$\mathcal{ALCHOIQ}^{\mathsf{Disj}}_{\mathsf{Self}}$ KB*. Step 1 is polynomial in $\mathcal{K}$, and so is 2 assuming that in each $\geqslant m\, s.B$, m is encoded in unary. For full $\mathcal{SROIQ}$ step 3 may result in an exponentially larger KB, but it remains polynomial for Horn-$\mathcal{SHOIQ}$.

Proposition 1. *For every Horn-$\mathcal{SROIQ}$ KB $\mathcal{K}$, there is a normal Horn-$\mathcal{ALCHOIQ}^{\mathsf{Disj}}_{\mathsf{Self}}$ KB $\mathcal{K}'$ that is satisfiable iff $\mathcal{K}$ is. The size of $\mathcal{K}'$ is bounded by an exponential function in the size of $\mathcal{K}$, and $\mathcal{K}'$ can be effectively constructed from $\mathcal{K}$ in single exponential time. If $\mathcal{K}$ is in Horn-$\mathcal{SHOIQ}$, then the size of $\mathcal{K}'$ and the time required to construct it are polynomially bounded (assuming that numbers are encoded unary).*

Deciding KB satisfiability

In this section we provide a method to decide the satisfiability of a given normal Horn-$\mathcal{ALCHOIQ}^{\mathsf{Disj}}_{\mathsf{Self}}$ KB $\mathcal{K}$ by a polynomial reduction to DATALOG. The language of DATALOG that we employ is formally defined next.

[1] RIAs and role assertions are usually considered to constitute a third component called RBox; we consider them part of the TBox instead. The other role assertions usually allowed in $\mathcal{SROIQ}$ are expressible in the usual way, cf. (Horrocks, Kutz, & Sattler 2006).

DATALOG$^{\mathrm{S}}$

We introduce an extension of standard DATALOG that we call DATALOG$^{\mathrm{S}}$. It supports *set sorts* that facilitate the manipulation of sets, allowing for a simpler description of the reduction. We will see below that DATALOG$^{\mathrm{S}}$ is not more expressive and can be rewritten into the standard one.

Definition 1 (DATALOG$^{\mathrm{S}}$)**.** Fix a countably infinite set C of constants. We call C *the element sort*. Given any finite $S \subset \mathsf{C}$, 2^S is called an S-sort (and a *set sort* if the particular S is not relevant). The element sort and all set sorts are *sorts*. Let $\mathsf{N}_{\mathsf{Rel}}$ be a countably infinite set of *relation names*, where each $\mathsf{R} \in \mathsf{N}_{\mathsf{Rel}}$ is associated to an arity $n(\mathsf{R}) \geq 0$. Each $\mathsf{R} \in \mathsf{N}_{\mathsf{Rel}}$ is assigned a *sort function* σ_{R} that maps each position $0 < i \leq n(\mathsf{R})$ to a sort. The *sort signature* of R is $\Sigma(\mathsf{R}) = \{S | \exists i \in \{1, \ldots, n(\mathsf{R})\} : \sigma_{\mathsf{R}}(i) \text{ is an } S\text{-sort}\}$.

For each sort $\mathcal{S}$, let $\mathsf{V}^{\mathcal{S}}$ be a countably infinite set of *$\mathcal{S}$-variables*. Then each $t \in \mathsf{V}^{\mathcal{S}} \cup \mathcal{S}$ is an $\mathcal{S}$-term.[2] Additionally, if $\mathcal{S}$ is a set sort and t_1 and t_2 are $\mathcal{S}$-terms, then $t_1 \sqcup t_2$ is an $\mathcal{S}$-term. Then an *atom* is an expression $\mathsf{R}(t_1, \ldots, t_n)$, where for each $1 \leq i \leq n(\mathsf{R})$, t_i is an $\sigma_{\mathsf{R}}(i)$-term.

A *rule* is an expression R of the form $b_1, \ldots, b_n \rightarrow h$, where $\{b_1, \ldots, b_n\}$ is a possibly empty set of *body* atoms, and h is a *head* atom or blank. If h is blank, R is a *constraint*. For each rule R we require *safety*: each variable occurring in a head atom of R also occurs in some body atom of R. Rules with no body atoms are *facts*, all other rules are *intensional rules*. A *program* is a finite set of rules. The *sort signature* $\Sigma(P)$ of P is the union of $\Sigma(\mathsf{R})$ over all relations R of P. A term (resp., atom, rule, program) is *ground* if it contains no variables. A ground rule R' is a *ground instance* of a rule R if R' can be obtained from R by replacing each $\mathcal{S}$-variable occurring in R by some ground term $t \in \mathcal{S}$.

An *interpretation* I for a program P maps each n-ary relation R of P to a set $\mathsf{R}^I \subseteq \sigma_{\mathsf{R}}(1) \times \cdots \times \sigma_{\mathsf{R}}(n)$. The *evaluation* $e(t)$ of a ground $\mathcal{S}$-term t is defined as follows: (i) $e(t) = t$ if $t \in \mathcal{S}$, and (ii) $e(t) = e(t_1) \cup e(t_2)$ if $t = t_1 \sqcup t_2$. A ground atom $\mathsf{R}(t_1, \ldots, t_n)$ is *true in* I if $\langle e(t_1), \ldots, e(t_n) \rangle \in \mathsf{R}^I$. *$I$ satisfies a ground constraint* r, if r contains a body atom that is false in I. For all remaining ground rules r, *I satisfies* r if the head atom of r is true in I whenever all body atoms of r are true in I. I is a *model* of a program P if I satisfies each ground instance of each rule in P. P is consistent if it has model. Given two interpretations I and J for P, we write $J \prec I$ if (i) for each relation R of P, we have $\mathsf{R}^J \subseteq \mathsf{R}^I$, and (ii) for some relation R the inclusion is strict. A model I of P is *minimal* if there exists no model J of P such that $J \prec I$.

A (standard) DATALOG program P is a DATALOG$^{\mathrm{S}}$ program with $\Sigma(P) = \emptyset$. As in the standard case, every consistent DATALOG$^{\mathrm{S}}$ program P has a unique minimal model, called the *least model* of P.

Example 1. The following DATALOG$^{\mathrm{S}}$ program P, where all arguments have sort 2^S (i.e., are of S-sort), defines the subset relation over 2^S:

$$\begin{array}{rcl} & \rightarrow & \mathsf{Set}(S), \\ \mathsf{Set}(X \sqcup Y) & \rightarrow & \mathsf{Set}(X), \\ \mathsf{Set}(X \sqcup Y) & \rightarrow & \mathsf{SubsetOf}(X, X \sqcup Y). \end{array}$$

Indeed, for any $t_1, t_2 \subseteq S$, if I is the least model model of P then $\mathsf{SubsetOf}(t_1, t_2)$ is true in I iff $t_1 \subseteq t_2$.

Given the above (polynomial) axiomatization of the subset relation for an S-sort, we can w.l.o.g. assume the availability in DATALOG$^{\mathrm{S}}$ of a 'built-in' $\subseteq$ predicate, which we use in infix notation. We further use $c \in t$ as an abbreviation for $\{c\} \subseteq t$, and use $\cup$ in place of $\sqcup$ in what follows.

Recall that consistency in plain DATALOG is EXPTIME-complete. If the intensional rules are fixed, i.e., the case of data complexity, the problem drops to PTIME-completeness (see e.g. (Dantsin *et al.* 2001)). The presence of set sorts in DATALOG$^{\mathrm{S}}$ does not increase the complexity: via a polynomial translation into DATALOG, which is given in the appendix, we obtain the following result.

Theorem 2. *Testing consistency of a* DATALOG$^{\mathrm{S}}$ *program P is feasible in exponential time in the combined size of P $\Sigma(P)$. If the intensional rules and the sort signature $\Sigma(P)$ of P are fixed, checking consistency of P is feasible in polynomial time in the size of P.*

Reduction to DATALOG$^{\mathrm{S}}$

Assume a normal Horn-$\mathcal{ALCHOIQ}^{\mathsf{Disj}}_{\mathsf{Self}}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$. We present here a (polynomial) satisfiability preserving translation of $\mathcal{K}$ into a DATALOG$^{\mathrm{S}}$ program $P_{\mathcal{K}}$, which can in turn be polynomially converted into plain DATALOG.

The challenge is the following: $\mathcal{K}$ might have infinite (and only infinite) models, while models of DATALOG programs are always finite relational structures. For this reason, we do not use rules to build models of $\mathcal{K}$. Instead, we use rules to derive the different combinations of concepts (*types*) and the different relations between types that *must appear* in every model of $\mathcal{K}$. If no contradictions are encountered in this derivation, then a model of $\mathcal{K}$ exists and it can be assembled from the derived types and relations. We exploit the fact that $\mathcal{K}$ is Horn: by explicating the relevant relations that must appear in *every* model of $\mathcal{K}$, we obtain one single *universal* model whenever $\mathcal{K}$ is satisfiable. Furthermore, due to the absence of disjunction, this derivation can be carried out deterministically using DATALOG.

Definition 2. We denote by $\mathsf{N}_{\mathsf{R}}(\mathcal{K})$ (resp., $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$, $\mathsf{N}_{\mathsf{I}}(\mathcal{T})$) the set of all role names (resp., individuals) occurring in $\mathcal{K}$ (resp., $\mathcal{T}$), and we let $\overline{\mathsf{N}_{\mathsf{R}}}(\mathcal{K}) = \mathsf{N}_{\mathsf{R}}(\mathcal{K}) \cup \{\bar{r} \mid r \in \mathsf{N}_{\mathsf{R}}(\mathcal{K})\}$. $\mathsf{C}(\mathcal{K})$ denotes the following set of concepts: (1) if $C \in \mathsf{N}_{\mathsf{C}}$ or $C = \{a\}$ for some $a \in \mathsf{N}_{\mathsf{I}}(\mathcal{T})$, and C occurs in $\mathcal{K}$, then $C \in \mathsf{C}(\mathcal{K})$, and (2) $\exists r.\mathrm{Self} \in \mathsf{C}(\mathcal{K})$ for every $r \in \mathsf{N}_{\mathsf{R}}(\mathcal{K})$. A *type* is any set $\tau \subseteq \mathsf{C}(\mathcal{K})$. For each element d in the domain of an interpretation $\mathcal{I}$, we say that d *realizes a type* τ, if $d \in \bigcap_{C \in \tau} C^{\mathcal{I}}$, i.e. d satisfies each concept in τ.[3]

For the encoding, we view roles, individuals, and concepts as constants. To manipulate types and sets of roles,

[2] By allowing the use *sets of constants* as (syntactic) terms, we tacitly assume that they have a suitable syntactic representation.

[3] Note that τ only needs to be contained in the set of concepts C such that $d \in C^{\mathcal{I}}$, but not to be equal to it.

we use two set sorts: argument positions of $\mathsf{C}(\mathcal{K})$-sort and $\overline{\mathsf{N_R}}(\mathcal{K})$-sort are used for subsets of $\mathsf{C}(\mathcal{K})$ (i.e. types) and subsets of $\overline{\mathsf{N_R}}(\mathcal{K})$, respectively. In the following we use (possibly subscripted) variables T, R, X for types, sets of roles and individuals, respectively.

The rules of $P_\mathcal{K}$ define several relations over individuals, types, and sets of roles. In particular they define two relations called realized and enf. The former will store the types realized in models of $\mathcal{K}$, while enf will provide triples $\langle \tau_1, \rho, \tau_2 \rangle$ which, intuitively, read as follows: if an element d realizes τ_1, then it must have a successor d' such that d and d' are related by the roles in ρ, and d' realizes τ_2. As we shall see, $\mathcal{K}$ is satisfiable iff a model can be built using the types and sets of roles in these relations.

We now provide the rules of $P_\mathcal{K}$, which define realized, enf, and several auxiliary relations.

Basic implications. First we introduce the binary relation imp over types: intuitively, $\mathsf{imp}(\tau_1, \tau_2)$ is true iff τ_1 *implies* τ_2, i.e., if in every model of $\mathcal{K}$, every d that realizes τ_1 also realizes τ_2.

Since imp must capture all concept implications that arise from the TBox axioms, we initialize it with a ground fact for each axioms of type $A \sqcap B \sqsubseteq C$ in $\mathcal{T}$ as follows:

$$\rightarrow \mathsf{imp}(\{A, B\}, \{C\}). \tag{1}$$

The imp relation must be *implicational* in the sense that it is closed under the following rules:

$$\begin{aligned} \rightarrow \mathsf{imp}(\{\}, \{\top\}), \qquad & \rightarrow \mathsf{imp}(\{\bot\}, \mathsf{C}(\mathcal{K})), \\ T_1 \subseteq T_2 \ & \rightarrow \mathsf{imp}(T_2, T_1), \\ \mathsf{imp}(T_1, T_2), \mathsf{imp}(T_2, T_3) \ & \rightarrow \mathsf{imp}(T_1, T_3), \\ \text{and, for each } C \in \mathsf{C}(\mathcal{K}), & \\ \mathsf{imp}(T_1, T_2) \ & \rightarrow \mathsf{imp}(T_1 \cup \{C\}, T_2 \cup \{C\}). \end{aligned} \tag{2}$$

Enforced relations. The enf predicate is a ternary relation with sorts $\langle \mathsf{C}(\mathcal{K}), \overline{\mathsf{N_R}}(\mathcal{K}), \mathsf{C}(\mathcal{K}) \rangle$. Formally, a type τ_1 *enforces* a ρ-successor τ_2, if in every model $\mathcal{I}$ of $\mathcal{K}$, for each domain element d that realizes τ_1, there exists an element d' realizing τ_2 such that $\langle d, d' \rangle \in \bigcap_{r \in \rho} r^\mathcal{I}$.

We initialize enf with the following fact for each axiom of type $A \sqsubseteq \exists r.B$ in $\mathcal{T}$:

$$\rightarrow \mathsf{enf}(\{A\}, \{r\}, \{B\}). \tag{3}$$

Further, if τ_1 enforces a ρ-successor τ_2, then any τ_1' that implies τ_1 also enforces an ρ-successor realizing τ_2, and every τ_2' implied by τ_2. This is ensured with the following rule:

$$\mathsf{enf}(T_1, R, T_2), \mathsf{imp}(T_1', T_1), \mathsf{imp}(T_2, T_2') \rightarrow \mathsf{enf}(T_1', R, T_2') \tag{4}$$

To ensure satisfiability of $\mathcal{K}$, we require that for every tuple in the enf relation, ρ is compatible with the RIs in $\mathcal{T}$. This is achieved with the following rules for each $r \sqsubseteq s$ in $\mathcal{T}$:

$$\begin{aligned} \mathsf{enf}(T_1, R, T_2), r \in R \ & \rightarrow \mathsf{enf}(T_1, R \cup \{s\}, T_2) \\ \mathsf{enf}(T_1, R, T_2), \bar{r} \in R \ & \rightarrow \mathsf{enf}(T_1, R \cup \{\bar{s}\}, T_2). \end{aligned} \tag{5}$$

Also, if the enf relation contains a tuple $\langle \tau_1, \rho, \tau_2 \rangle$ where ρ violates some disjointness axiom, then the type τ_1 causes an inconsistency. This is reflected in the following rules for each axiom $\mathsf{Disj}(r, s)$ in $\mathcal{T}$:

$$\begin{aligned} \mathsf{enf}(T_1, R, T_2), \{r, s\} \subseteq R \ & \rightarrow \mathsf{imp}(T_1, \{\bot\}), \\ \mathsf{enf}(T_1, R, T_2), \{\bar{r}, \bar{s}\} \subseteq R \ & \rightarrow \mathsf{imp}(T_1, \{\bot\}). \end{aligned} \tag{6}$$

The effect of GCIs of the form $A \sqsubseteq \forall r.B$ is also captured in the enf relation. First, if some τ enforces a ρ-successor that realizes τ_2, and we have $A \in \tau$ and $r \in \rho$, then the successor must also satisfy B. On the other hand, if ρ contains the inverse of r and $A \in \tau_2$, then each element realizing τ_1 must also satisfy B. Hence, for all axioms of type $A \sqsubseteq \forall r.B$, we have:

$$\begin{aligned} &\mathsf{enf}(T_1, R, T_2), A \in T_1, r \in R \rightarrow \mathsf{enf}(T_1, R, T_2 \cup \{B\}), \\ &\mathsf{enf}(T_1, R, T_2), \bar{r} \in R, A \in T_2 \rightarrow \mathsf{imp}(T_1, \{B\}). \end{aligned} \tag{7}$$

For handling the number restrictions we set up a new predicate inv that holds between a set of roles and its role-wise inverse. This is easily done using the next two rules for each $r \in \overline{\mathsf{N_R}}(\mathcal{K})$:

$$\begin{aligned} &\rightarrow \mathsf{inv}(\{r\}, \{\bar{r}\}), \\ \mathsf{inv}(R_1, R_2) &\rightarrow \mathsf{inv}(R_1 \cup \{r\}, R_2 \cup \{\bar{r}\}). \end{aligned} \tag{8}$$

We deal with the axioms $A \sqsubseteq \leq 1r.C$ by appropriately merging types and sets of roles. We have to consider two situations. First, if a type enforces several successor types, then they must be merged if the number restrictions apply. This is done by adding for each $A \sqsubseteq \leq 1r.C \in \mathcal{T}$ the rule:

$$\begin{aligned} \mathsf{enf}(T, R_1, T_1), &\mathsf{enf}(T, R_2, T_2), \\ A \in T, r \in R_1, r \in R_2, & \\ C \in T_1, C \in T_2 &\rightarrow \mathsf{enf}(T, R_1 \cup R_2, T_1 \cup T_2) \end{aligned} \tag{9}$$

The second, slightly more complicated situation, is the following. Assume an axiom $A \sqsubseteq \leq 1r.C$, and assume that some type τ_2 with $A \in \tau_2$ enforces a ρ_2-successor τ_3 with $r \in \rho_2$ and $C \in \tau_3$. Suppose also that τ_2 is itself enforced as an ρ_1-successor of some τ_1 with $\bar{r} \in \rho_1$ and $C \in \tau_1$. In this case we need to 'merge' τ_1 and τ_3, along with the corresponding roles. I.e. any domain element realizing τ_1 must also realize τ_3, and the roles relating τ_1 and τ_2 must be augmented with the role-wise inverse of ρ_2. This is captured by the following pair of rules for each $A \sqsubseteq \leq 1r.C$ in $\mathcal{T}$:

$$\begin{aligned} \mathsf{enf}(T_1, R_1, T_2), &\mathsf{enf}(T_2, R_2, T_3), \\ A \in T_2, r \in R_2, \bar{r} \in R_1, & \\ C \in T_1, C \in T_3, \mathsf{inv}(R_2, R_2') &\rightarrow H \end{aligned} \tag{10}$$

for $H = \mathsf{imp}(T_1, T_3)$ and $\mathsf{enf}(T_1, R_1 \cup R_2', T_2)$.

Dealing with Self-loops. So far we have treated concepts of the form $\exists r.\mathrm{Self}$ in the same way as concept names. Now we make sure that they are given the right semantics. First, every τ that contains some $\exists r.\mathrm{Self}$ concept enforces itself as successor type:

$$\{\exists r.\mathrm{Self}\} \in T \ \rightarrow \ \mathsf{enf}(T, \{r\}, T). \tag{11}$$

Self 'loops' must also be compatible with the disjointness axioms, role inclusions, and the universal restrictions. In what follows, we will use $\hat{r}$ to denote r if the role r is in $\mathsf{N_R}$, and $\bar{r}$ otherwise. For each axiom $\mathsf{Disj}(r, s)$, $r \sqsubseteq s$, and $A \sqsubseteq \forall r.B$ of $\mathcal{T}$ we have a rule (12), (13) and (14), respectively:

$$\{\exists \hat{r}.\mathrm{Self}, \exists \hat{s}.\mathrm{Self}\} \in T \rightarrow \mathsf{imp}(T, \{\bot\}), \tag{12}$$

$$\rightarrow \mathsf{imp}(\{\exists \hat{r}.\mathrm{Self}\}, \{\exists \hat{s}.\mathrm{Self}\}), \tag{13}$$

$$\rightarrow \mathsf{imp}(\{A, \exists \hat{r}.\mathrm{Self}\}, \{B\}). \tag{14}$$

To capture the effect of number restrictions in the presence of $\exists r.\mathrm{Self}$ concepts, we add the following rules for each

$A \sqsubseteq \leq 1r.C \in \mathcal{T}$ and each $r' \in \overline{\mathsf{N_R}}(\mathcal{K})$:

$$\begin{array}{r}\mathsf{enf}(T_1,R,T_2),r\in R,\\ C\in T_2,\{A,\exists\hat{r}.\mathrm{Self}\}\subseteq T_1\rightarrow\mathsf{imp}(T_1,T_2),\\ \mathsf{enf}(T_1,R,T_2),\{\bar{r},r'\}\subseteq R,\\ C\in T_1,\{A,\exists\hat{r}.\mathrm{Self},C\}\subseteq T_2,\rightarrow\mathsf{imp}(T_1,\{\exists r'.\mathrm{Self}\}),\\ \mathsf{enf}(T_1,R,T_2),\{r,r'\}\subseteq R\\ C\in T_2,\{A,\exists\hat{r}.\mathrm{Self}\}\subseteq T_1\rightarrow\mathsf{imp}(T_1,\{\exists r'.\mathrm{Self}\}).\end{array} \tag{15}$$

Nominals. There are two important things about nominals. First, if $\{a\}$ is a nominal in $\mathcal{T}$, then each model of $\mathcal{K}$ must realize some type τ with $\{a\}\in\tau$. On the other hand, in each model of $\mathcal{K}$ there is a unique domain element satisfying $\{a\}$.

As already mentioned, we use the unary realized relation to store the realized types. Formally, a type τ *is realized* if in any model $\mathcal{I}$ of $\mathcal{K}$ there exists an element that realizes τ. Then the realization of types containing nominals is initiated by the following fact for each $\{a\}\in\mathsf{C}(\mathcal{K})$:

$$\rightarrow\ \mathsf{realized}(\{\{a\}\}) \tag{16}$$

If a type τ is realized, then all the types that it implies and that it enforces are also realized:

$$\begin{array}{rcl}\mathsf{realized}(T_1),\mathsf{enf}(T_1,R,T_2)&\rightarrow&\mathsf{realized}(T_2)\\ \mathsf{realized}(T_1),\mathsf{imp}(T_1,T_2)&\rightarrow&\mathsf{realized}(T_2)\end{array} \tag{17}$$

Further, inconsistent types containing $\{\bot\}$ may not be realized anywhere in the models of $\mathcal{K}$:

$$\mathsf{realized}(\{\bot\})\rightarrow. \tag{18}$$

To ensure the uniqueness of a domain element satisfying a nominal, we use a binary same relation over types. Intuitively, if $\mathsf{same}(\tau_1,\tau_2)$ is true, then we must identify any pair of elements d_1 and d_2 satisfying τ_1 and τ_2, respectively. For each $\{a\}\in\mathsf{C}(\mathcal{K})$, we add:

$$\begin{array}{r}\mathsf{realized}(T_1),\mathsf{realized}(T_2),\\ \{a\}\in T_1,\{a\}\in T_2\rightarrow\mathsf{same}(T_1,T_2).\end{array} \tag{19}$$

Clearly, pairs of types in the same relation imply each other, and their union is also realized:

$$\begin{array}{l}\mathsf{same}(T_1,T_2)\rightarrow\mathsf{imp}(T_1,T_2),\\ \mathsf{same}(T_1,T_2)\rightarrow\mathsf{realized}(T_1\cup T_2).\end{array} \tag{20}$$

Tuples in same may witness the existence of self-loops and enforced relations:

$$\begin{array}{l}\mathsf{enf}(T_1,\{r\},T_2),\mathsf{same}(T_1,T_2)\rightarrow\mathsf{imp}(T_1,\{\exists\hat{r}.\mathrm{Self}\}),\\ \mathsf{enf}(T_1,R_1,T_3),\mathsf{enf}(T_2,R_2,T_4),\\ \quad\mathsf{same}(T_1,T_2),\mathsf{same}(T_3,T_4)\rightarrow\mathsf{enf}(T_1,R_1\cup R_2,T_3).\end{array} \tag{21}$$

We must also take care of the relations to nominals: in the presence of number restrictions, elements that have relations to nominals might need to be identified (e.g. if they both have some nominal as r successor, and the nominal can only have one incoming r arc). In fact, whole sequences of domain elements leading to a nominal might need to be 'collapsed' into one single sequence of *pseudo nonimals*. This is captured by propagating same over the enf relation. For all axiom $A\sqsubseteq\leq 1r.C\in\mathcal{T}$ we add:

$$\begin{array}{r}\mathsf{realized}(T_1),\mathsf{enf}(T_1,R_1,T_3),\\ \mathsf{realized}(T_2),\mathsf{enf}(T_2,R_2,T_4),\\ C\in T_1,C\in T_2,\\ \bar{r}\in R_1,\bar{r}\in R_2,A\in T_3,\\ \mathsf{same}(T_3,T_4)\rightarrow\mathsf{same}(T_1,T_2)\end{array} \tag{22}$$

This finishes the encoding for the TBox $\mathcal{T}$. Since nominals can be used to simulate ABoxes, this encoding could already be used as a procedure for KB satisfiability. However, in order to obtain optimal data-complexity results, we add rules that handle the ABox explicitely.

Encoding the ABox Assertions. For handling the concept assertions in $\mathcal{A}$, we define a binary relation realizes over individuals and types. Intuitively, $\mathsf{realizes}(a,\tau)$ means that the element interpreting a must realize τ. For all assertions $a:A$ and all individuals $a\in\mathsf{N_I}(\mathcal{T})$, we add the following facts:

$$\rightarrow\mathsf{realizes}(a,\{A\}), \tag{23}$$

$$\rightarrow\mathsf{realizes}(a,\{a\}). \tag{24}$$

The next rules capture general properties of realization:

$$\begin{array}{r}\mathsf{realizes}(X,T_1),\mathsf{imp}(T_1,T_2)\rightarrow\mathsf{realizes}(X,T_2)\\ \mathsf{realizes}(X,T_1),\mathsf{realizes}(X,T_2)\rightarrow\mathsf{realizes}(X,T_1\cup T_2)\\ \mathsf{realizes}(X,T)\rightarrow\mathsf{realized}(T).\end{array} \tag{25}$$

For the role assertions we proceed similarly. We define a ternary predicate realize over individuals and sets of roles: realize provides us the triples $\langle a,b,\rho\rangle$ such that in any model $\mathcal{I}$ of $\mathcal{K}$, $\langle a^{\mathcal{I}},b^{\mathcal{I}}\rangle\in r^{\mathcal{I}}$ for all $r\in\rho$. We add the following fact for each role assertion (a,b):r in $\mathcal{T}$:

$$\rightarrow\mathsf{realize}(a,b,\{r\}). \tag{26}$$

For each $r\in\overline{\mathsf{N_R}}(\mathcal{K})$, we have a rule to explicate the inverse relations, and a rule for aggregating realized role sets:

$$\mathsf{realize}(X_1,X_2,R),r\in R\rightarrow\mathsf{realize}(X_2,X_1,\{\bar{r}\}), \tag{27}$$

$$\begin{array}{r}\mathsf{realize}(X_1,X_2,\rho_1),\\ \mathsf{realize}(X_1,X_2,\rho_2)\ \rightarrow\mathsf{realize}(X_1,X_2,\rho_1\cup\rho_2)\end{array} \tag{28}$$

In the realize relation we must also reflect the connections stemming from enforced relations to (pseudo) nominals. This is formalized by following rule:

$$\begin{array}{r}\mathsf{realizes}(X_1,T_1),\mathsf{enf}(T_1,R,T_2),\\ \mathsf{same}(T_2,T_2),\mathsf{realizes}(X_2,T_2)\rightarrow\mathsf{realize}(X_1,X_2,R)\end{array} \tag{29}$$

To ensure that the meaning of $\exists r.\mathrm{Self}$ concepts is reflected in the realize relation, we add, for each $r\in\mathsf{N_R}(\mathcal{K})$:

$$\begin{array}{rcl}\mathsf{realize}(X_1,X_1,\{r\})&\rightarrow&\mathsf{realizes}(X_1,\{\exists r.\mathrm{Self}\})\\ \mathsf{realizes}(X_1,\{\exists r.\mathrm{Self}\})&\rightarrow&\mathsf{realize}(X_1,X_1,\{r\})\end{array} \tag{30}$$

To make sure that the relations between ABox individuals do not violate any axiom in $\mathcal{T}$, we add a rule of the form (31) for each $r\sqsubseteq s$; a rule (32) for each $A\sqsubseteq\forall r.B$; and a rules (33) for each $\mathsf{Disj}(r,s)$:

$$\mathsf{realize}(X_1,X_2,\{r\})\rightarrow\mathsf{realize}(X_1,X_2,\{s\}) \tag{31}$$

$$\begin{array}{r}\mathsf{realize}(X_1,X_2,R),r\in R,\\ \mathsf{realizes}(X_1,T),A\in T\rightarrow\mathsf{realizes}(X_2,\{B\})\end{array} \tag{32}$$

$$\mathsf{realize}(X_1,X_2,R),\{r,s\}\subseteq R\rightarrow \tag{33}$$

Ensuring the satisfaction of the axioms $A\sqsubseteq\leq 1r.C$ is more involved. To this aim, we axiomatize a congruence relation equal between ABox individuals as follows:

$$\begin{array}{r}\text{for each } a\in\mathsf{N_I}(\mathcal{K}),\quad\rightarrow\mathsf{equal}(a,a),\\ \mathsf{equal}(X_1,X_2)\rightarrow\mathsf{equal}(X_2,X_1),\\ \mathsf{equal}(X_1,X_2),\mathsf{equal}(X_2,X_3)\rightarrow\mathsf{equal}(X_1,X_3),\\ \mathsf{realizes}(X_1,T_1),\mathsf{realizes}(X_2,T_2),\\ \mathsf{same}(T_1,T_2)\rightarrow\mathsf{equal}(X_1,X_2),\\ \mathsf{realizes}(X_1,T),\mathsf{equal}(X_1,X_2)\rightarrow\mathsf{realizes}(X_2,T),\\ \mathsf{realize}(X_1,X_2,R),\mathsf{equal}(X_1,X_3)\rightarrow\mathsf{realize}(X_3,X_2,R),\\ \mathsf{realize}(X_1,X_2,R),\mathsf{equal}(X_2,X_3)\rightarrow\mathsf{realize}(X_1,X_3,R).\end{array} \tag{34}$$

Now we can ensure the satisfaction of the number restrictions by adding, for each $A \sqsubseteq \leq 1r.C$, rules of the form:

$$\begin{array}{r} \mathsf{realizes}(X_1, T_1), A \in T_1, \\ \mathsf{realize}(X_1, X_2, R_1), r \in R_1, \\ \mathsf{realize}(X_1, X_3, R_2), r \in R_2, \\ \mathsf{realizes}(X_2, T_2), C \in T_2, \\ \mathsf{realizes}(X_3, T_3), C \in T_3 \ \rightarrow \ \mathsf{equal}(X_2, X_3), \end{array} \tag{35}$$

$$\begin{array}{r} \mathsf{realize}(X_1, T_1), A \in T_1, \\ \mathsf{realize}(X_1, X_2, R_1), r \in R_1, \\ \mathsf{realized}(T_2), C \in T_2, \mathsf{enf}(T_2, R_2, T_1), \bar{r} \in R_2, \\ \mathsf{realizes}(X_2, \{C\}), \mathsf{same}(T_1, T_1), \mathsf{inv}(R_2, R_2') \rightarrow H, \end{array} \tag{36}$$

$$\begin{array}{r} \mathsf{realize}(X_1, T_1), A \in T_1, \\ \mathsf{realize}(X_1, X_2, R_1), r \in R_1, \\ \mathsf{enf}(T_1, R_2, T_2), r \in R_2 \\ \mathsf{realizes}(X_2, \{C\}), C \in T_2 \rightarrow H'. \end{array} \tag{37}$$

for $H = \mathsf{realize}(X_1, X_2, R_2')$ and $H = \mathsf{realizes}(X_2, T_2)$, and for $H' = \mathsf{realize}(X_1, X_2, R_1 \cup R_2)$ and $H' = \mathsf{realizes}(X_1, T_2)$.

These ensure the satisfaction of atmost restrictions by identifying ABox individuals (35) and merging enforced types into ABox individuals if required (36 and 37).

This finishes the encoding of $\mathcal{K}$ into the DATALOG$^{\neg s}$ program $P_{\mathcal{K}}$. Now we prove that the reduction is correct.

Proposition 3. *$\mathcal{K}$ is satisfiable iff $P_{\mathcal{K}}$ has a model.*

Proof. (Sketch) For the *only if* direction, we have to show that from a model $\mathcal{I}$ of $\mathcal{K}$ we can construct an interpretation I for $P_{\mathcal{K}}$ that satisfies all the rules above. The construction of I is straightforward and reflects the intuitive meaning of predicates we have described. For example, imp^I contains the pairs of types τ, τ' such that $\bigcap_{A \in \tau} A^{\mathcal{I}} \subseteq \bigcap_{B \in \tau'} B^{\mathcal{I}}$, while enf^I contains the triples $\langle \tau, \rho, \tau' \rangle$ such that for every d realizing τ there is a d' realizing τ' such that $\langle d, d' \rangle \in \bigcap_{r \in \rho} r^{\mathcal{I}}$. The interpretation of realized contains all realized types, i.e. the set of τ such that $\bigcap_{A \in \tau} A^{\mathcal{I}} \neq \emptyset$. For the same predicate, we simply put all pairs of types such that there is a unique $d \in \Delta^{\mathcal{I}}$ that realizes simultaneously τ and τ', i.e., $\langle \tau, \tau' \rangle \in \mathsf{same}^I$ iff $\bigcap_{A \in \tau} A^{\mathcal{I}} = \bigcap_{B \in \tau'} B^{\mathcal{I}} = \{d\}$ for some $d \in \Delta^{\mathcal{I}}$. A pair of individuals a, b occurring in $\mathcal{K}$ is in equal^I iff $a^{\mathcal{I}} = b^{\mathcal{I}}$, while $\langle a, \tau \rangle \in \mathsf{realizes}^I$ iff τ is realized by $a^{\mathcal{I}}$; $\mathsf{realize}^I$ is analogous. Other auxiliary predicates are trivial. It is easy to confirm the satisfaction of rules (1) to (37).

The *if* direction is more interesting, and it is proved by showing that the least model of $P_{\mathcal{K}}$ gives rise to a model $\mathcal{I}$ of $\mathcal{K}$. The domain $\Delta^{\mathcal{I}}$ of $\mathcal{I}$ has a forest-like structure, and we allow each element to be connected to the forest roots (apart from its predecessor, its children and itself.) As roots of $\Delta^{\mathcal{I}}$ we use sets of individuals (induced as equivalence classes of the equal relation) and additional elements called *pseudo-nominals*. Each pseudo-nominal satisfies a set of types that are pairwise identified via the same predicate. To grow a tree out of every root, we create new successors as demanded by enf. Roughly we ignore triples in enf which are not maximal (w.r.t. second and third argument) and hence subsumed by others. For each node d realizing some τ and each maximal $\langle \tau, \rho, \tau' \rangle$ it enforces, we create a new successor, only if the respective ρ and τ' are not already satisfied by taking the predecessor of d or d itself; if τ' is in the same relation then d is connected to a pseudo-nominal and no new successor is created. With these strategies we obtain a model where each element has sufficient neigbours to satisfy all axioms, and no more neighbours than the number restrictions allow. Note that the rules dealing with atmost restrictions enforce the existence of the desired unique maximal enf triples. To show that the constructed forest is a model, one must examine all the relevant axiom types from Table 1. A detailed proof is given in the appendix. □

Complexity of Reasoning

From the encoding of Horn-$\mathcal{SHOIQ}$ into DATALOG$^{\neg s}$ and Theorem 2, we obtain the following complexity results:

Theorem 4. *The satisfiability problem for Horn-$\mathcal{SHOIQ}$ and Horn-$\mathcal{SROIQ}$ KBs is in* EXPTIME *and* 2-EXPTIME*, respectively (in the former case, we assume unary encoding of numbers). If the TBox is fixed, i.e. when measuring data complexity, the problem for both DLs is in* PTIME*.*

Proof. Given a Horn-$\mathcal{SHOIQ}$ KB, the translation into normal Horn-$\mathcal{ALCHOIQ}^{\mathsf{Disj}}_{\mathsf{Self}}$ is polynomial, and so is the translation into DATALOG$^{\neg s}$ that was given in the previous section. For Horn-$\mathcal{SROIQ}$ the first step may result in a single-exponential blowup, and the second step remains polynomial. The respective EXPTIME and 2-EXPTIME upper bounds are then obtained using Theorem 2.

For the data complexity, observe that fixing the TBox of a KB means fixing all the intensional rules and the sort signature of its DATALOG$^{\neg s}$ translation. Indeed, each $a : A$ in the ABox translates into facts $\mathsf{realizes}(a, \{A\})$ and $\mathsf{equal}(a, a)$, while each $(a, b) : r$ translates into $\mathsf{realize}(a, b, \{r\})$, $\mathsf{equal}(a, a)$ and $\mathsf{equal}(b, b)$); these do not alter the sort signature. Hence we obtain from Theorem 2 that satisfiability testing is feasible in polynomial time. □

These bounds are tight. KB satisfiability in any DL allowing for conjunction on the left hand side and quantified universal restrictions on the right hand side of TBox axioms is PTIME-hard in data complexity (Calvanese *et al.* 2006). EXPTIME-hardness in combined complexity holds already for Horn-$\mathcal{SHIQ}$ (Krötzsch, Rudolph, & Hitzler 2007), a fragment of Horn-$\mathcal{SHOIQ}$.

2-EXPTIME hardness of satisfiability of Horn-$\mathcal{SROIQ}$ KBs can be shown by a reduction from the word problem for a deterministic Turing machine with double-exponentially bounded time. The construction, which closely follows (Kazakov 2008), is in the appendix. Thus we obtain:

Theorem 5. *Deciding the satisfiability of a Horn-$\mathcal{SROIQ}$ knowledge base is* 2-EXPTIME*-hard.*

Discussion and Conclusion

In this paper, we presented an encoding of Horn-$\mathcal{SHOIQ}$ and Horn-$\mathcal{SROIQ}$ KBs into DATALOG, and obtained this way optimal complexity bounds for the two DLs. The reduction is based on *types* and hence follows some of the ideas already exploited for other FOL fragments (cf. (Grädel & Otto 1999)).

A different encoding into DATALOG was developed in (Hustadt, Motik, & Sattler 2004; 2005) for $\mathcal{SHIQ}$ and

Horn-$\mathcal{SHIQ}$. There, the authors apply a resolution procedure to obtain a DATALOG program that can be evaluated in polynomial or nondeterministic polynomial time, depending on the presence of disjunction. Due to the EXPTIME-hardness of these logics, it might take exponential time to compute the DATALOG program itself. Contrarily, our translation is polynomial, but the resulting program might take exponential time to evaluate.

Our approach is also related to the one in (Kazakov 2009) for Horn-$\mathcal{SHIQ}$, which works by explicating implicit inclusion axioms. We do similar inferences in the DATALOG program: the imp and enf relations explicate dependencies between types, which can be viewed as new TBox axioms.

Our translation is modular, and works on an axiom-by-axiom basis. If the signature of the KB is fixed, updates can be easily incorporated. If new assertions or axioms are added to the KB, the least model of the new DATALOG program can be built on top of the model of the old one.

Many issues remain for future work. For example, conjunctive query answering in the two logics could be attempted by exploiting the type representation of a canonical model given by the least model of the DATALOG program.

References

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P. 2007. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press.

Baader, F. 2003. Terminological cycles in a description logic with existential restrictions. In *Proc. IJCAI'03*, 325–330.

Calvanese, D.; Giacomo, G. D.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2006. Data complexity of query answering in description logics. In *Proc. KR'06*, 260–270.

Calvanese, D.; Giacomo, G. D.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning* 39(3):385–429.

Dantsin, E.; Eiter, T.; Gottlob, G.; and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Computing Surveys* 33(3):374–425.

Grädel, E., and Otto, M. 1999. On Logics with Two Variables. *J. Theor. Comput. Sci.* 224(1-2):73–113.

Horrocks, I., and Sattler, U. 2005. A Tableaux Decision Procedure for SHOIQ. In *Proc. IJCAI'05*, 448–453.

Horrocks, I.; Kutz, O.; and Sattler, U. 2006. The Even More Irresistible $\mathcal{SROIQ}$. In *Proc. KR'06*, 57–67. AAAI Press.

Hustadt, U.; Motik, B.; and Sattler, U. 2004. Reducing SHIQ-Description Logic to Disjunctive Datalog Programs. In *Proc. KR'04*, 152–162. AAAI Press.

Hustadt, U.; Motik, B.; and Sattler, U. 2005. Data complexity of reasoning in very expressive description logics. In *Proc. IJCAI'05*, 466–471. Professional Book Center.

Kazakov, Y., and Motik, B. 2008. A Resolution-based Decision Procedure for *SHOIQ*. *J. Autom. Reasoning* 40(2-3):89–116.

Kazakov, Y. 2008. $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are Harder than $\mathcal{SHOIQ}$. In *Proc. KR'08*, 274–284. AAAI Press.

Kazakov, Y. 2009. Consequence-driven Reasoning For *Horn-SHIQ* Ontologies. In *Proc. IJCAI'09*, 2040–2045.

Krötzsch, M.; Rudolph, S.; and Hitzler, P. 2007. Complexity Boundaries for *Horn* Description Logics. In *Proc. AAAI'07*.

Rudolph, S.; Krötzsch, M.; and Hitzler, P. 2008. Terminological reasoning in $\mathcal{SHIQ}$ with ordered binary decision diagrams. In *Proc. AAAI'08*, 529–534. AAAI Press.

Tobies, S. 2000. The Complexity of Reasoning with Cardinality Restrictions and Nominals in Expressive Description Logics. *J. Artif. Intell. Res. (JAIR)* 12:199–217.

Appendix

From DATALOG$^{\mathrm{S}}$ to DATALOG

Assume a DATALOG$^{\mathrm{S}}$ program P with sort signature $\Sigma(P)$. To translate P into plain DATALOG, we need to eliminate set positions and set variables from P. This can done by employing a bit-vector representation of sets, i.e., for each $S \in \Sigma(P)$, each subset of S is represented as a vector of $|S|$ bits. For each $S \in \Sigma(P)$ we assume an arbitrary but fixed enumeration of its elements, and, with a slight abuse of notation, write $S(i)$ to denote the ith element in S.

The transformation is then defined in 3 steps:

1. (Elimination of $\cup$) For each $S \in \Sigma(P)$, we use a new ternary relation symbol U_S not occurring in P. In every rule $r \in P$, we replace each term $t_1 \cup t_2$ by a fresh variable X, and add to the body of r the atom $\mathsf{U}_S(t_1, t_2, X)$. In this way P becomes $\cup$-free.

2. (Atom rewriting) For the $\cup$-free P we construct a plain DATALOG program P'. We additionally use two fresh constants 0 and 1. P' is obtained from P by replacing each atom $R(t_1, \ldots, t_n)$ in P by $R'(tr(t_1), \ldots, tr(t_n))$, where $tr(t)$ is defined as follows:

(a) if $t = X$ and $X \in \mathsf{V}^{\mathsf{C}}$, i.e. X is a variable for individual constants, then $tr(t) = X'$;

(b) if $t = X$ and X is a variable for the S-sort, then $tr(X) = \langle X'_{S(1)}, \ldots, X'_{S(m)} \rangle$, where $m = |S|$.

(c) if $t = \{c_1, \ldots, c_k\}$ is a ground term for the S-sort, then $tr(t) = \langle b_1, \ldots, b_m \rangle$, where $m = |S|$ and for each $1 \leq i \leq m$: (i) $b_i = 1$ if $S(i) \in t$, and (ii) $b_i = 0$, otherwise.

3. (Dealing U_S) We finally need to ensure that the relation U'_S which results from U_S behaves like the union predicate: $\mathsf{U}'_S(\vec{X}, \vec{Y}, \vec{Z})$ must be true if $\vec{Z}$ encodes the union of the two sets encoded in $\vec{X}$ and $\vec{Y}$. This is achieved by adding to P' the following rules for each $S \in \Sigma(P)$, where $m = |S|$:

$$\mathsf{max}(0,0,0);\ \mathsf{max}(1,0,1);\ \mathsf{max}(0,1,1);\ \mathsf{max}(1,1,1);$$
$$\mathsf{max}(X_1, Y_1, Z_1), \ldots, \mathsf{max}(X_m, Y_m, Z_m) \rightarrow \mathsf{U}'_S(\vec{X}, \vec{Y}, \vec{Z}).$$

It is easy to establish a correspondence between models of P and P'. We note that the resulting program is of polynomial size in the size of P and $\Sigma(P)$. Observe also that in case the sort signature $\Sigma(P)$ is fixed, adding new facts does not require extending predicate arities, and that a fresh ground fact for P translates into a fresh new fact for P'. This justifies the claims in Theorem 2.

Proof of Proposition 3, *if* direction

Recall that $\mathcal{K}$ is a normal Horn-$\mathcal{ALCHOIQ}^{\mathsf{Disj}}_{\mathsf{Self}}$ knowledge base, and $P_{\mathcal{K}}$ denotes its translation into DATALOG$^{\mathrm{S}}$.

Definition 3. Let I be the least model of $P_{\mathcal{K}}$. We define the *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *induced by* I as follows.

Recall that equal^I is an equivalence relation over the individuals in $\mathsf{N_I}(\mathcal{K})$. We define the set $\mathfrak{R}$ of *roots* as the set of all elements of the form $\mathbf{c}\tau$ for which:

- $\mathbf{c}$ is either an equivalence class of equal^I, or the symbol $\diamond$;
- if $\mathbf{c} \neq \diamond$, then τ is the $\subseteq$-minimal concept type containing each τ' such that $\langle a, \tau' \rangle \in \mathsf{realizes}^I$ for some $a \in \mathbf{c}$; and
- if $\mathbf{c} = \diamond$ then (i) $\langle \tau, \tau \rangle \in \mathsf{same}^I$, (ii) there is no $\tau' \supset \tau$ with $\langle \tau', \tau' \rangle \in \mathsf{same}^I$, and (iii) there is no $a \in \mathsf{N_I}(\mathcal{K})$ with $\langle a, \tau \rangle \in \mathsf{realizes}^I$.

We define the set $\mathfrak{O} \subseteq \mathfrak{R}$ of *pseudo-nominals* as $\{\mathbf{c}\tau \in \mathfrak{R} \mid \langle \tau, \tau \rangle \in \mathsf{same}^I\}$. Given a concept type τ, the set $\mathrm{ES}(\tau)$ of its *enforced successors* contains the pairs $\langle \rho_1, \tau_1 \rangle \in 2^{\mathsf{C}(\mathcal{K})} \times 2^{\mathsf{N_R}(\mathcal{K})}$ for which $\langle \tau, \rho_1, \tau_1 \rangle \in \mathsf{enf}^I$ and there is no $\langle \tau, \rho_2, \tau_2 \rangle \in \mathsf{enf}^I$ with $\rho_1 \cup \tau_1 \subset \rho_2 \cup \tau_2$.

In the following, we will work on words d of the shape $d = \mathbf{c}\tau_1\rho_1\tau_2 \dots \rho_{n-1}\tau_n \in \mathfrak{R} \times (2^{\mathsf{N_R}(\mathcal{K})} \times 2^{\mathsf{C}(\mathcal{K})})^*$, and we use $[\![d]\!]$ to denote the last element τ_n of d.

The domain $\Delta^{\mathcal{I}}$ is the smallest set of words such that $\mathfrak{R} \subseteq \Delta^{\mathcal{I}}$, and if $d \in \Delta^{\mathcal{I}}$ and $\langle \rho, \tau \rangle \in \mathrm{ES}([\![d]\!])$ then $d\rho\tau \in \Delta^{\mathcal{I}}$ unless one of the following is the case:

E1 there is a $\mathbf{c}\tau \in \mathfrak{O}$,
E2 $\{\exists \hat{r}.\mathrm{Self} \mid r \in \rho\} \cup \tau \subseteq [\![d]\!]$,
E3 $d = d'\rho'\tau'$ with $\{\bar{r} \mid r \in \rho\} \subseteq \rho'$ and $\tau \subseteq [\![d']\!]$,
E4 $d = \mathbf{c}\tau' \in \mathfrak{O}$ and there is a $\tau'' \in \mathsf{realized}^I$ with $\langle \rho', \tau' \rangle \in \mathrm{ES}(\tau'')$ such that $\{\bar{r} \mid r \in \rho\} \subseteq \rho'$ and $\tau \subseteq \tau''$.
E5 $d = \mathbf{c}\tau' \in \mathfrak{R}$, $\mathbf{c} \neq \diamond$, and there are $\langle a, b, \rho' \rangle \in \mathsf{realize}^I$ and $\mathbf{c}'\tau'' \in \mathfrak{R}$ with $a \in \mathbf{c}$, $b \in \mathbf{c}'$, $\rho \subseteq \rho'$ and $\tau \subseteq \tau''$.

We define the interpretation function $\cdot^{\mathcal{I}}$ as follows:

- for each $a \in \mathsf{N_I}(\mathcal{K})$, $a^{\mathcal{I}} = \mathbf{c}\tau$ for the $\mathbf{c}\tau \in \mathfrak{R}$ with $a \in \mathbf{c}$;
- for each $A \in \mathsf{N_C}$ occurring in $\mathcal{K}$, $d \in A^{\mathcal{I}}$ iff $A \in [\![d]\!]$;
- for each $r \in \mathsf{N_R}(\mathcal{K})$, $r^{\mathcal{I}}$ is the set of pairs that contains:
  - R1 $\langle \mathbf{c}\tau, \mathbf{c}'\tau' \rangle$ whenever $\mathbf{c}\tau, \mathbf{c}'\tau' \in \Delta^{\mathcal{I}}$ and there are individuals $a \in \mathbf{c}$ and $b \in \mathbf{c}'$ with $\langle a, b, \{r\} \rangle \in \mathsf{realize}^I$,
  - R2 $\langle d, d\rho\tau \rangle$ whenever $d, d\rho\tau \in \Delta^{\mathcal{I}}$ and $r \in \rho$,
  - R3 $\langle d\rho\tau, d \rangle$ whenever $d, d\rho\tau \in \Delta^{\mathcal{I}}$ and $\bar{r} \in \rho$,
  - R4 $\langle d, d \rangle$ whenever $\exists r.\mathrm{Self} \in [\![d]\!]$,
  - R5 $\langle d, d' \rangle$ if $d' \in \mathfrak{O}$ and $\langle \rho, [\![d']\!] \rangle \in \mathrm{ES}([\![d]\!])$ for a $\rho \ni r$,
  - R6 $\langle d', d \rangle$ if $d' \in \mathfrak{O}$ and $\langle \rho, [\![d']\!] \rangle \in \mathrm{ES}([\![d]\!])$ for a $\rho \ni \mathsf{Inv}(r)$.

To show $\mathcal{I} \models \mathcal{K}$, we first prove some auxiliary lemmas:

Lemma 6. *$\tau \in \mathsf{realized}^I$ iff $\tau \subseteq [\![d]\!]$ for some $d \in \Delta^{\mathcal{I}}$.*

Proof. The "if" part can be proven inductively via the construction of $\Delta^{\mathcal{I}}$. For the base case, consider $d = \mathbf{c}\tau$. If $\mathbf{c} \neq \diamond$ then $\langle a, \tau' \rangle \in \mathsf{realizes}^I$ for some $a \in \mathbf{c}$ and Rule 25.3 ensure $\tau \in \mathsf{realized}^I$. If $\mathbf{c} = \diamond$, then $\langle \tau, \tau \rangle \in \mathsf{same}^I$ and Rule 20.2 ensures $\tau \in \mathsf{realized}^I$. For the induction step, consider a $d = d'\rho'\tau'$. By definition $\langle [\![d']\!], \rho', \tau' \rangle \in \mathsf{enf}^I$. As $[\![d']\!] \in \mathsf{realized}^I$ by induction hypothesis, Rule 17.1 implies $\tau' \in \mathsf{realized}^I$. The "only if" part then follows directly from the fact that I is the least model of $P_{\mathcal{K}}$. □

Lemma 7. *$C^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid C \in [\![d]\!]\}$ for each $C \in \mathsf{C}(\mathcal{K})$.*

Proof. If C is a concept name in $\mathsf{N_C}$, the claim holds immediately by definition of $C^{\mathcal{I}}$. For $C = \{a\}$ with $a \in \mathsf{N_I}(\mathcal{T})$ it follows from the Rules 24 and 34.4. For $C = \exists r.\mathrm{Self}$, $C \in [\![d]\!]$ implies $d \in C^{\mathcal{I}}$ by the definition of $r^{\mathcal{I}}$ and we only need to show $\subseteq$. We consider an arbitrary $d \in (\exists r.\mathrm{Self})^{\mathcal{I}}$. As $\langle d, d \rangle \in r^{\mathcal{I}}$, we have case R1, R4, R5 or R6 in the definition of $r^{\mathcal{I}}$. If R1, then d is of the form $c\tau$ and $\{\exists r.\mathrm{Self}\} \in \tau$ is ensured by Rules 30 and 34. Case R4 is trivial. For R5 and R6, $d = d' \in \mathfrak{O}$ implies $\langle [\![d]\!], [\![d]\!] \rangle \in \mathsf{same}^I$ and hence Rule 21.1 ensures $\{\exists r.\mathrm{Self}\} \in [\![d]\!]$. □

Lemma 8. *For every concept type τ with $\langle \tau, \tau \rangle \in \mathsf{same}^I$ there is exactly one $d \in \Delta^{\mathcal{I}}$ that realizes τ.*

Proof. By definition of $\mathfrak{O}$ it is obvious that there is at least one such d. Moreover, by definition and via E1, we know that every such d must be of the form $\mathbf{c}\tau'$. Assume $d_1 = \mathbf{c}_1\tau_1$ and $d_2 = \mathbf{c}_2\tau_2$ with $\tau \subseteq \tau_1$ and $\tau \subseteq \tau_2$. Now, as I is the least model, there must be a derivation justifying $\langle \tau, \tau \rangle \in \mathsf{same}^I$. An inspection of the rules for same reveals that this derivation directly gives rise to one for $\langle \tau_1, \tau_2 \rangle \in \mathsf{same}^I$. Hence $\tau_1 \cup \tau_2 \in \mathsf{realized}^I$ by Rule 20, and we can similarly see that there is a derivation of $\langle \tau_1 \cup \tau_2, \tau_1 \cup \tau_2 \rangle \in \mathsf{same}^I$.

Now suppose $\mathbf{c}_1 = \diamond$, which by definition holds iff $\mathbf{c}_2 = \diamond$ as well. The definition requires τ_1 and τ_2 to be maximal, so we have $\tau_1 = \tau_2 = \tau_1 \cup \tau_2$, which implies $d_1 = d_2$. Otherwise, if $\mathbf{c}_1 \neq \diamond$ and $\mathbf{c}_2 \neq \diamond$, let $a \in \mathbf{c}_1$ and $b \in \mathbf{c}_2$. Then by Rule 25,$\langle a, \tau_1 \rangle \in \mathsf{realizes}^I$ as well as $\langle b, \tau_2 \rangle \in \mathsf{realizes}^I$. As we already found $\langle \tau_1, \tau_2 \rangle \in \mathsf{same}^I$, we can use Rule 34.4 to obtain $\langle a, b \rangle \in \mathsf{equal}^I$ and hence by definition $\mathbf{c}_1 = \mathbf{c}_2$. This entails $\tau_1 = \tau_2$ and thus $d_1 = d_2$ as well. □

Lemma 9. *Let $d \in \Delta^{\mathcal{I}}$, and let τ_1 and τ_2 be concept types. If $\tau_1 \subseteq [\![d]\!]$ and $\langle \tau_1, \tau_2 \rangle \in \mathsf{imp}^I$, then $\tau_2 \subseteq [\![d]\!]$.*

Proof. We distinguish three cases: For $d = d'\rho\tau'$, we can use Rule 4 to deduce $\tau_2 \in \tau'$. For $d = \mathbf{c}\tau$ with $\mathbf{c} \neq \diamond$, $\tau_2 \subseteq \tau$ is ensured by Rule 25.1. It remains to consider $d = \diamond\tau$. Note that $\tau \in \mathsf{realized}^I$ (by Lemma 6) and $\langle \tau_1, \tau_2 \rangle \in \mathsf{imp}^I$ together with Rule 17.2 yield $\tau \cup \tau_2 \in \mathsf{realized}^I$. Yet from Lemma 8 it follows that $\diamond\tau$ is the only element realizing τ, which together with Lemma 6 entails $\tau_2 \subseteq \tau$. □

We now show the *if* direction of Proposition 3.

Lemma 10. *If the least model of $P_{\mathcal{K}}$ induces $\mathcal{I}$, then $\mathcal{I} \models \mathcal{K}$.*

Proof. We examine every axiom of $\mathcal{K}$ according to its type from Table 1 and show that it is satisfied in $\mathcal{I}$.

(i),(ii')▶ $r \sqsubseteq s \in \mathcal{T}$. Let $\langle d, d' \rangle \in r^{\mathcal{I}}$. We distinguish cases according to the reasons for which the pair $\langle d, d' \rangle$ is in $r^{\mathcal{I}}$:

R1 $d = \mathbf{c}\tau$,$d' = \mathbf{c}'\tau'$ and $\langle a, b, \{r\} \rangle \in \mathsf{realize}^I$ with $a \in \mathbf{c}$ and $b \in \mathbf{c}'$. Then Rule 31 ensures $\langle a, b, \{s\} \rangle \in \mathsf{realize}^I$ as well and therefore $\langle d, d' \rangle \in s^{\mathcal{I}}$.
R2 $\langle d, d\rho\tau \rangle$ with $r \in \rho$. We have $\langle [\![d]\!], \rho, \tau \rangle \in \mathsf{enf}^I$ by construction. We obtain $\langle [\![d]\!], \rho \cup \{s\}, \tau \rangle \in \mathsf{enf}^I$ by Rule 5.1. Via maximality we have $s \in \rho$, hence $\langle d, d' \rangle \in s^{\mathcal{I}}$.
R3 We argue the same way, using Rule 5.2 instead of 5.1.
R4 $d = d'$ with $\exists r.\mathsf{Self} \in [\![d]\!]$. Then, by Rule 13, and Lemma 9 we obtain $\exists s.\mathsf{Self} \in [\![d]\!]$.
R5 and R6 are analog to Cases R2 and R3.

(vi)▶ $\mathsf{Disj}(r, s) \in \mathcal{T}$. Assume towards a contradiction that $\langle d, d' \rangle \in r^{\mathcal{I}}$ and $\langle d, d' \rangle \in s^{\mathcal{I}}$. Again we examine all possible cases in which the pair is in $r^{\mathcal{I}}$:

R1 $d = \mathbf{c}\tau$, $d' = \mathbf{c}'\tau'$. We consider all cases with $\langle a, b, \{r\}\rangle \in \mathsf{realize}^I$ and $a \in \mathbf{c}$ and $b \in \mathbf{c}'$. All further combinations are subsumed by these cases or follow by symmetry. Now we examine the cases which cause $\langle d, d'\rangle \in s^{\mathcal{I}}$:

By R1. $\langle a', b', \{r\}\rangle \in \mathsf{realize}^I$ with $a' \in \mathbf{c}$ and $b' \in \mathbf{c}'$ $\langle a', b', \{s\}\rangle \in \mathsf{realize}^I$. By construction of $\mathbf{c}$ and $\mathbf{c}'$, we obtain $\langle a, a'\rangle \in \mathsf{equal}^I$ and $\langle b, b'\rangle \in \mathsf{equal}^I$, hence by Rules 34, we have $\langle a, b, \{s\}\rangle \in \mathsf{realize}^I$ and then, by Rule 28 $\langle a, b, \{r, s\}\rangle \in \mathsf{realize}^I$. This is a contradiction by Rule 33.

By R4. $d = d'$ with $\exists s.\mathrm{Self} \in [\![d]\!]$. Then, by construction, we have $\langle a, b\rangle \in \mathsf{equal}^I$ and by Rules 34 $\langle a, a, \{r\}\rangle \in \mathsf{realize}^I$. On the other hand, we get $\langle a, a, \{s\}\rangle \in \mathsf{realize}^I$ by Rule 30. Then Rule 28 ensures $\langle a, a, \{r, s\}\rangle \in \mathsf{realize}^I$, which contradicts Rule 33.

By R5. $d' \in \mathfrak{O}$ and $\langle \rho, [\![d']\!]\rangle \in \mathrm{ES}([\![d]\!])$ for some ρ with $s \in \rho$. Then Rule 29 entails $\langle a, b, \rho\rangle \in \mathsf{realize}^I$, such that we can apply Rule 28 to get $\langle a, b, \rho \cup \{s\}\rangle \in \mathsf{realize}^I$. This contradicts Rule 33.

By R6. $d \in \mathfrak{O}$ and $\langle \rho, [\![d]\!]\rangle \in \mathrm{ES}([\![d']\!])$ for some ρ with $\bar{s} \in \rho$ Then Rule 29 entails $\langle b, a, \rho\rangle \in \mathsf{realize}^I$, hence we get $\langle a, b, \{\bar{r'} \mid r' \in \rho\}\rangle \in \mathsf{realize}^I$ by Rule 27. Now, Rule 28 yields $\langle a, b, \{\bar{r'} \mid r' \in \rho\} \cup \{s\}\rangle \in \mathsf{realize}^I$. Again, this contradicts Rule 33.

R2 $d' = d\rho\tau$ with $\{r, s\} \in \rho$ Then, by construction, we have $\langle [\![d]\!], \rho, \tau\rangle \in \mathsf{enf}^I$ and can apply Rule 6.1 to obtain $\langle [\![d]\!], \{\bot\}\rangle \in \mathsf{imp}^I$ and therefore (noting Lemma 6) by Rules 17 $\{\bot\} \in \mathsf{realized}^I$. This contradicts Rule 18.

R3 $d = d'\rho\tau$ with $\{\bar{r}, \bar{s}\} \in \rho$. $\langle [\![d']\!], \rho, \tau\rangle \in \mathsf{enf}^I$ by construction. $\langle [\![d']\!], \{\bot\}\rangle \in \mathsf{imp}^I$ by Rule 6.2. By Lemma 6 and Rules 17 $\{\bot\} \in \mathsf{realized}^I$, contradicting Rule 18.

R4 $d = d'$ with $\{\exists r.\mathrm{Self}, \exists s.\mathrm{Self}\} \subseteq [\![d]\!]$. Then, by Rule 12, we obtain $\langle [\![d]\!], \{\bot\}\rangle \in \mathsf{imp}^I$ whence (noting Lemma 6) by Rules 17 $\{\bot\} \in \mathsf{realized}^I$. Once more, this contradicts Rule 18.

R5 and R6 are analog to Cases R2 and R3.

(vii)▶ $A \sqcap B \sqsubseteq C \in \mathcal{T}$. By Rules 1 and 2, we obtain $\langle \tau \cup \{A, B\}, \tau \cup \{A, B, C\}\rangle \in \mathsf{imp}^I$ for every $\tau \in \mathsf{C}(\mathcal{K})$. Given a $d \in A^{\mathcal{I}} \cap B^{\mathcal{I}}$ we find by Lemma 7 $A, B \in [\![d]\!]$ and by Lemma 9 conclude $C \in [\![d]\!]$. We finally invoke Lemma 7 to conclude $d \in C^{\mathcal{I}}$. In the following, we will tacitly presume Lemma 7 and directly argue via the value of $[\![\cdot]\!]$.

(viii)▶ $A \sqsubseteq \forall r.B \in \mathcal{T}$. Suppose $d, d' \in \Delta^{\mathcal{I}}$ with $d \in A^{\mathcal{I}}$ and $\langle d, d'\rangle \in r^{\mathcal{I}}$. We consider the causes for $\langle d, d'\rangle \in r^{\mathcal{I}}$:

R1 $d = \mathbf{c}\tau, d' = \mathbf{c}'\tau'$ with $\langle a, b, \{r\}\rangle \in \mathsf{realize}^I$ with $a \in \mathbf{c}$ and $b \in \mathbf{c}'$. Then we have $\langle a, \{A\}\rangle \in \mathsf{realizes}^I$, thus Rule 32 ensures $\langle b, \{B\}\rangle \in \mathsf{realizes}^I$ and thus $b^{\mathcal{I}} \in B^{\mathcal{I}}$.

R2 $d' = d\rho\tau$ and $r \in \rho$. By construction of $\Delta^{\mathcal{I}}$, we have $\langle [\![d]\!], \rho, \tau\rangle \in \mathsf{enf}^I$. Rule 7.1 yields $\langle [\![d]\!], \rho, \tau \cup \{B\}\rangle \in \mathsf{enf}^I$. Due to $\langle [\![d]\!], \rho, \tau\rangle$ being inclusion-maximal w.r.t. ρ and τ, we get $B \in \tau$ and $d' \in B^{\mathcal{I}}$.

R3 $d = d'\rho\tau$ and $\mathsf{Inv}(r) \in \rho$. Again, by construction of $\Delta^{\mathcal{I}}$, we have $\langle [\![d']\!], \rho, \tau\rangle \in \mathsf{enf}^I$. From $A \in \tau$ and $\mathsf{Inv}(r) \in \rho$, we can use Rule 7.2 to infer $\langle [\![d']\!], \{B\}\rangle \in \mathsf{imp}^I$ and then apply Lemma 9 to obtain $B \in [\![d']\!]$.

R4 $d = d'$ and $\exists r.\mathrm{Self} \in [\![d]\!]$. Then applying Rule 14 followed by Lemma 9 establishes $B \in [\![d]\!]$.

R5 and R6 are analog to Cases R2 and R3.

(ix)▶ $A \sqsubseteq \exists r.B \in \mathcal{T}$. Consider a $d \in A^{\mathcal{I}}$. By Lemma 7 we have $A \in [\![d]\!]$. By Rule 3, we obtain $\langle \{A\}, \{r\}, \{B\}\rangle \in \mathsf{enf}^I$. Then there is a $\langle \rho, \tau\rangle \in \mathrm{ES}([\![d]\!])$ with $r \in \rho$ and $B \in \tau$. By definition of $\Delta^{\mathcal{I}}$, if none of the exceptional cases applies, we have $d\rho\tau \in \Delta^{\mathcal{I}}$, and by R2 it follows that $\langle d, d\rho\tau\rangle \in r^{\mathcal{I}}$ and $B \in \tau = [\![d\rho\tau]\!]$. Now, we go through the exceptional cases, one by one:

E1 If there is a $\mathbf{c}\tau \in \mathfrak{O}$, we get $\langle d, \mathbf{c}\tau\rangle \in r^{\mathcal{I}}$ by R5, moreover $B \in [\![\mathbf{c}\tau]\!]$.

E2 If $\{\exists r.\mathrm{Self} \mid r \in \rho\} \cup \tau \subseteq [\![d]\!]$, then by R4 also $\langle d, d\rangle \in r^{\mathcal{I}}$ and $B \in \tau \subseteq [\![d]\!]$.

E3 If $d = d'\rho'\tau'$ with $\{\bar{r} \mid r \in \rho\} \subseteq \rho'$ and $\tau \subseteq [\![d']\!]$, we have $\langle d, d'\rangle \in r^{\mathcal{I}}$ and $B \in \tau \subseteq [\![d']\!]$

E4 $d = \mathbf{c}\tau' \in \mathfrak{O}$ and there is a $\tau'' \in \mathsf{realized}^I$ with $\langle \rho', \tau'\rangle \in \mathrm{ES}(\tau'')$ such that $\{\mathsf{Inv}(r) \mid r \in \rho\} \subseteq \rho'$ and $\tau \subseteq \tau''$. From $\tau'' \in \mathsf{realized}^I$ and Lemma 6, we obtain that there must be a $d' \in \Delta^{\mathcal{I}}$ with $[\![d']\!] = \tau''$. Then, by R6, we get $\langle d, d'\rangle \in r^{\mathcal{I}}$ and $B \in \tau \subseteq \tau'' \subseteq [\![d']\!]$.

E5 $d = \mathbf{c}\tau' \in \mathfrak{R}$ with $\mathbf{c} \neq \diamond$ and there are $\langle a, b, \rho'\rangle \in \mathsf{realize}^I$ and $\mathbf{c}'\tau'' \in \mathfrak{R}$ with $a \in \mathbf{c}$, $b \in \mathbf{c}'$, $\rho \subseteq \rho'$ and $\tau \subseteq \tau''$. Then, by R1, we know $\langle d, \mathbf{c}'\tau''\rangle \in r^{\mathcal{I}}$ and $B \in \tau \subseteq \tau'' = [\![\mathbf{c}'\tau'']\!]$.

(xi)▶ $A \sqsubseteq \leqslant 1\, r.C \in \mathcal{T}$. Suppose there are d, d_1, d_2 with $A \in [\![d]\!]$, $\langle d, d_i\rangle \in r^{\mathcal{I}}$ for $i \in \{1, 2\}$, $C \in [\![d_1]\!]$ and $C \in [\![d_2]\!]$. We examine all combinations of cases, and show $d_1 = d_2$ for all those that can result in the two pairs being in $r^{\mathcal{I}}$:

R1 and R1: $d = \mathbf{c}\tau, d_1 = \mathbf{c}_1\tau_1, d_2 = \mathbf{c}_2\tau_2$, $\langle a, b, \{r\}\rangle \in \mathsf{realize}^I$ with $a \in \mathbf{c}$ and $b \in \mathbf{c}_1$, $\langle a', b', \{r\}\rangle \in \mathsf{realize}^I$ with $a' \in \mathbf{c}$ and $b' \in \mathbf{c}_2$. By construction of $\mathbf{c}$, we obtain $\langle a, a'\rangle \in \mathsf{equal}^I$, hence by Rule 34, we have $\langle a, b', \{r\}\rangle \in \mathsf{realize}^I$ and then, by Rule 35, get $\langle b, b'\rangle \in \mathsf{equal}^I$. Then, by construction, $\mathbf{c}_1 = \mathbf{c}_2$ and therefore $d_1 = d_2$.

R1 and R2: $d = \mathbf{c}\tau, d_1 = \mathbf{c}_1\tau_1$, $\langle a, b, \{r\}\rangle \in \mathsf{realize}^I$ with $a \in \mathbf{c}$ and $b \in \mathbf{c}_1$, $d_2 = d\rho_2\tau_2$ with $r \in \rho_2$. Since $\langle [\![d]\!], \rho_2, \tau_2\rangle \in \mathsf{enf}^I$, we can apply Rule 37 to obtain $\langle b, \tau_2\rangle \in \mathsf{realizes}^I$ and $\langle a, b, \rho_2\rangle \in \mathsf{realize}^I$. But then $d\rho_2\tau_2$ cannot occur in Δ^I due to E5.

R1 and R3: This case cannot occur, as d can not be of both forms $d = \mathbf{c}\tau$ and $d = d_2\rho_2\tau_2$.

R1 and R4: $d = \mathbf{c}\tau, d_1 = \mathbf{c}_1\tau_1$, $\langle a, b, \{r\}\rangle \in \mathsf{realize}^I$ with $a \in \mathbf{c}$ and $b \in \mathbf{c}_1$, $d_2 = d$, $\exists r.\mathrm{Self} \in [\![d]\!]$. By Rule 30, we obtain $\langle a, a, \{r\}\rangle \in \mathsf{realize}^I$ and consequently by Rule 35 $\langle a, b\rangle \in \mathsf{equal}^I$, thus $\mathbf{c}_1 = \mathbf{c}$ and hence $d_1 = d = d_2$.

R1 and R5: $d = \mathbf{c}\tau, d_1 = \mathbf{c}_1\tau_1$, $\langle a, b, \{r\}\rangle \in \mathsf{realize}^I$ with $a \in \mathbf{c}$ and $b \in \mathbf{c}_1$, $d_2 \in \mathfrak{O}$ and $\langle \rho, [\![d_2]\!]\rangle \in \mathrm{ES}([\![d]\!])$ for some ρ with $r \in \rho$. Since $\langle [\![d]\!], \rho_2, [\![d_2]\!]\rangle \in \mathsf{enf}^I$, we can apply Rule 37 to obtain $\langle b, [\![d_2]\!]\rangle \in \mathsf{realizes}^I$. Then due to $d_2 \in \mathfrak{O}$ and Lemma 8, we obtain $d_1 = d_2$.

R1 and R6: $d = \mathbf{c}\tau$, $d_1 = \mathbf{c}_1\tau_1$, $\langle a, b, \{r\}\rangle \in \mathsf{realize}^I$ with $a \in \mathbf{c}$ and $b \in \mathbf{c}_1$, $d \in \mathfrak{O}$ and $\langle \rho, [\![d]\!]\rangle \in \mathrm{ES}([\![d_2]\!])$ for some ρ with $\bar{r} \in \rho$. Due to $\langle [\![d_2]\!], \rho, [\![d]\!]\rangle \in \mathsf{enf}^I$, and $\langle [\![d]\!], [\![d]\!]\rangle \in \mathsf{same}^I$, we can apply Rule 36 to obtain $\langle b, [\![d_2]\!]\rangle \in \mathsf{realizes}^I$ as well as Rule 22 to obtain $\langle [\![d_2]\!], [\![d_2]\!]\rangle \in \mathsf{same}^I$. This ensures $d_2 = \mathbf{c}_2\tau_2$ with $b \in \mathbf{c}_2$ and hence $d_1 = d_2$.

R2 and R2: $d_1 = d\rho_1\tau_1$, $r \in \rho_1$, $d_2 = d\rho_2\tau_2$, $r \in \rho_2$. Then we have $\langle [\![d]\!], \rho_1, \tau_1\rangle \in \mathsf{enf}^I$ and $\langle [\![d]\!], \rho_2, \tau_2\rangle \in$

enf^I, thus we get by Rule 10 together with 4 also $\langle [\![d]\!], \rho_1 \cup \rho_2, \tau_1 \cup \tau_2 \rangle \in \mathsf{enf}^I$. Hence, again by maximality, we know that $\rho_1 = \rho_2$ and $\tau_1 = \tau_2$ and thus $d_1 = d_2$.

R2 and R3: $d_1 = d\rho_1\tau_1$, $r \in \rho_1$, $d = d_2\rho_2\tau_2$, $\bar{r} \in \rho_2$. Then, $\langle \tau_2, \rho_1, \tau_1 \rangle \in \mathsf{enf}^I$ and $\langle [\![d_2]\!], \rho_2, \tau_2 \rangle \in \mathsf{enf}^I$ as well. By Rule , we then obtain $\langle [\![d_2]\!], \tau_1 \rangle \in \mathsf{imp}^I$ and $\langle [\![d_2]\!], \rho_2 \cup \{\bar{r} \mid r \in \rho_1\}, \tau_2 \rangle \in \mathsf{enf}^I$ and therefore via Rule 4 $\langle [\![d_2]\!], \rho_2 \cup \{\bar{r} \mid r \in \rho_1\}, [\![d_2]\!] \cup \tau_2 \rangle \in \mathsf{enf}^I$, which (by construction-maximality) entails $\{\bar{r} \mid r \in \rho_1\} \subseteq \rho_2$ and $\tau_1 \subseteq [\![d_2]\!]$. Then $d\rho_1\tau_1$ cannot be in $\Delta^{\mathcal{I}}$ as E3 would apply. Therefore this case cannot occur.

R2 and R4: $d_1 = d\rho_1\tau_1$, $r \in \rho_1$, $d_2 = d$, $\exists r.\mathsf{Self} \in [\![d]\!]$. Then, by Rules 15.1, 15.3, and 4, we know that $\{\exists \hat{r}.\mathrm{Self} \mid r \in \rho\} \cup \tau \subseteq [\![d]\!]$. But then $d\rho_1\tau_1$ cannot be in $\Delta^{\mathcal{I}}$ as E2 would apply. Thus this case cannot occur either.

R2 and R5: $d_1 = d\rho_1\tau_1$, $r \in \rho_1$, $d_2 \in \mathfrak{O}$ and $\langle \rho_2, [\![d_2]\!] \rangle \in \mathrm{ES}([\![d]\!])$ for some ρ_2 with $r \in \rho_2$. Then, $\langle [\![d]\!], \rho_1, \tau_1 \rangle \in \mathsf{enf}^I$ and $\langle [\![d]\!], \rho_2, \tau_2 \rangle \in \mathsf{enf}^I$ as well. By Rules 10 and 4, we obtain $\langle [\![d]\!], \rho_1 \cup \rho_2, \tau_1 \cup \tau_2 \rangle \in \mathsf{enf}^I$ and thus (by maximality) $\tau_1 = \tau_2$. But then $d\rho_1\tau_1$ cannot be in $\Delta^{\mathcal{I}}$ as E1 would apply. Hence, this case cannot occur.

R2 and R6: $d_1 = d\rho_1\tau_1$, $r \in \rho_1$, $d \in \mathfrak{O}$ and $\langle \rho_2, [\![d]\!] \rangle \in \mathrm{ES}([\![d_2]\!])$ for some ρ_2 with $\bar{r} \in \rho_2$. Then, $\langle [\![d]\!], \rho_1, \tau_1 \rangle \in \mathsf{enf}^I$ and $\langle [\![d_2]\!], \rho_2, [\![d]\!] \rangle \in \mathsf{enf}^I$ as well. Then, we get a contradiction analog to the case R2 and R3.

R3 and R3: $d = d_1\rho_1\tau_1$, $\bar{r} \in \rho_1$. $d = d_2\rho_2\tau_2$, $\bar{r} \in \rho_2$. By construction of $\Delta^{\mathcal{I}}$, we directly obtain $d_1 = d_2$.

R3 and R4: $d = d_1\rho_1\tau_1$, $\bar{r} \in \rho_1$, $d_2 = d$, $\exists r.\mathsf{Self} \in [\![d]\!]$. From these, we obtain by Rule 11 $\langle \tau_1, \{r\}, \tau_1 \rangle \in \mathsf{enf}^I$. As additionally $\langle [\![d_1]\!], \rho_1, \tau_1 \rangle \in \mathsf{enf}^I$, we can use Rule 10 to derive $\langle [\![d_1]\!], \tau_1 \rangle \in \mathsf{imp}^I$. But then, with Rules 4 and by maximality $\tau_1 \in [\![d_1]\!]$, moreover, by Rule 15.2 follows $\{\exists \hat{r}.\mathrm{Self} \mid r \in \rho_1\} \subseteq [\![d_1]\!]$. But then, $d_1\rho_1\tau_1$ cannot be in $\Delta^{\mathcal{I}}$ due to E2. So this case cannot occur.

R3 and R5: $d = d_1\rho_1\tau_1$, $\bar{r} \in \rho_1$, $d_2 \in \mathfrak{O}$ and $\langle \rho_2, [\![d_2]\!] \rangle \in \mathrm{ES}([\![d]\!])$ for some ρ_2 with $r \in \rho_2$. Then, $\langle [\![d_1]\!], \rho_1, \tau_1 \rangle \in \mathsf{enf}^I$ and $\langle \tau_1, \rho_2, [\![d_2]\!] \rangle \in \mathsf{enf}^I$ as well. By Rule , we obtain $\langle [\![d_1]\!], [\![d_2]\!] \rangle \in \mathsf{imp}^I$ and hence via Lemma 9 and maximality $[\![d_2]\!] = [\![d_1]\!]$. As $d_2 \in \mathfrak{O}$, we know by Lemma 8 that $[\![d_2]\!]$ can be realized only once, thus $d_1 = d_2$.

R3 and R6: $d = d_1\rho_1\tau_1$ and $d \in \mathfrak{O}$ cannot both be true.

R4 and R4: This case is obvious, as $d = d_1 = d_2$.

R4 and R5: $d_1 = d$, $\exists r.\mathsf{Self} \in [\![d]\!]$. $d_2 \in \mathfrak{O}$ and $\langle \rho_2, [\![d_2]\!] \rangle \in \mathrm{ES}([\![d]\!])$ for some ρ_2 with $r \in \rho_2$. Since $\langle [\![d]\!], \rho_2, [\![d_2]\!] \rangle \in \mathsf{enf}^I$, we can apply Rule 15.1 to obtain $\langle [\![d]\!], [\![d_2]\!] \rangle \in \mathsf{imp}^I$. Then, by Lemma 9 $[\![d_2]\!] \subseteq [\![d]\!]$ and as $d_2 \in \mathfrak{O}$, we can use Lemma 8 to obtain $d_1 = d = d_2$.

R4 and R6: $d_1 = d$, $\exists r.\mathsf{Self} \in [\![d]\!]$, $d \in \mathfrak{O}$ and $\langle \rho_2, [\![d]\!] \rangle \in \mathrm{ES}([\![d_2]\!])$ for some ρ_2 with $\bar{r} \in \rho_2$. From these, we obtain by Rule 11 $\langle [\![d]\!], \{r\}, [\![d]\!] \rangle \in \mathsf{enf}^I$. As additionally $\langle [\![d_2]\!], \rho_2, [\![d]\!] \rangle \in \mathsf{enf}^I$, Rule 10 yields $\langle [\![d_1]\!], [\![d]\!] \rangle \in \mathsf{imp}^I$. With Rules 4 and by maximality $[\![d]\!] \subseteq [\![d_2]\!]$. From $d \in \mathfrak{O}$ and Lemma 8 follows $d = d_1 = d_2$

R5 and R5: $d_1 \in \mathfrak{O}$ and $\langle \rho_1, [\![d_1]\!] \rangle \in \mathrm{ES}([\![d]\!])$ for some ρ_1 with $r \in \rho_1$, $d_2 \in \mathfrak{O}$ and $\langle \rho_2, [\![d_2]\!] \rangle \in \mathrm{ES}([\![d]\!])$ for some ρ_2 with $r \in \rho_2$. As we have $\langle [\![d]\!], \rho_1, [\![d_1]\!] \rangle \in \mathsf{enf}^I$ as well as $\langle [\![d]\!], \rho_2, [\![d_2]\!] \rangle \in \mathsf{enf}^I$, we get by Rules .1 and .2 together with 4 also $\langle [\![d]\!], \rho_1 \cup \rho_2, [\![d_1]\!] \cup [\![d_2]\!] \rangle \in \mathsf{enf}^I$. Hence, by maximality, we know that $[\![d_1]\!] = [\![d_2]\!]$ and therefore, via Lemma 8, $d_1 = d_2$.

R5 and R6: $d_1 \in \mathfrak{O}$ and $\langle \rho_1, [\![d_1]\!] \rangle \in \mathrm{ES}([\![d]\!])$ for some ρ_1 with $r \in \rho_1$, $d \in \mathfrak{O}$ and $\langle \rho_2, [\![d]\!] \rangle \in \mathrm{ES}([\![d_2]\!])$ for some ρ_2 with $\bar{r} \in \rho_2$. As in the case R5 and R6, we obtain $\langle [\![d_2]\!], [\![d_1]\!] \rangle \in \mathsf{imp}^I$, hence $[\![d_1]\!] \subseteq [\![d_2]\!]$, therefore (as $d_1 \in \mathfrak{O}$ and by Lemma 8) also $d_1 = d_2$.

R6 and R6: $d \in \mathfrak{O}$ and $\langle \rho_1, [\![d]\!] \rangle \in \mathrm{ES}([\![d_1]\!])$ for some ρ_1 with $\bar{r} \in \rho_1$, $\langle \rho_2, [\![d]\!] \rangle \in \mathrm{ES}([\![d_2]\!])$ for some ρ_2 with $\bar{r} \in \rho_2$. Then we have $\langle [\![d_1]\!], \rho_1, [\![d]\!] \rangle \in \mathsf{enf}^I$ and $\langle [\![d_2]\!], \rho_2, [\![d]\!] \rangle \in \mathsf{enf}^I$. Moreover, $d \in \mathfrak{O}$ implies $\langle [\![d]\!], [\![d]\!] \rangle \in \mathsf{same}^I$ such that we can apply Rule 22 to obtain $\langle [\![d_1]\!], [\![d_2]\!] \rangle \in \mathsf{same}^I$, whence via Rule 20, Lemma 9, and Lemma 8, we can conclude $d_1 = d_2$.

(xiii)▶ $a{:}A \in \mathcal{A}$. By Rule 23, $\langle a, \{A\} \rangle \in \mathsf{realizes}^I$, and by the definition of $\mathcal{I}$, we obtain $a^{\mathcal{I}} \in A^{\mathcal{I}}$ as desired.

(xiv)▶ $(a, b){:}r \in \mathcal{A}$. By Rule 26 we have $\langle a, b, \{r\} \rangle \in \mathsf{realize}^I$. Then the definition of $\mathcal{I}$ (in particular R1) yields $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in r^{\mathcal{I}}$. □

Horn-$\mathcal{SROIQ}$ is 2-ExpTime hard

We show 2-ExpTime hardness standard reasoning in Horn-$\mathcal{SROIQ}$ by reducing the word problem of a deterministic Turing machine that halts within double exponential time to the unsatisfiability of a Horn-$\mathcal{SROIQ}$ KB.

Axiomatizing a Double Exponential Grid First we show how to axiomatize a finite grid of size $2^{2^n} \times 2^{2^n}$ for a given n. This is done similarly as in (Kazakov 2008), but by introducing additional concept names to avoid using negated concepts, we obtain an axiomatization which is in Horn-$\mathcal{SROIQ}$.

We first use the well-known binary integer counting technique to axiomatize r-chains of length 2^n.

The ith low- and the ith high-bit cannot be set for the same element; every element of the r-chain having some low-bits set (and hence not being the 2^nth one) has an r-successor. The first bit is always flipped when going to the next element in the r-chain. Initial elements (those in class Z) have all low-bits set; we let E recognize the end of the r-chain (i.e. the individual with all high-bits set).

$$\begin{array}{cccc} L_i \sqcap H_i \sqsubseteq \bot & L_i \sqsubseteq \exists r.\top & 1 \le i \le n & \\ L_1 \sqsubseteq \forall r.H_1 & H_1 \sqsubseteq \forall r.L_1 & Z \sqsubseteq \bigsqcap_{i \le n} L_i & \bigsqcap_{i \le n} H_i \sqsubseteq E \end{array}$$

For every bit position $k > 1$, we introduce a marker S_k to indicate the bit with the largest position that should be flipped in the next element of the r-chain.

$$\begin{array}{ll} S_k \sqsubseteq \forall r.L_m \quad S_k \sqcap S_m \sqsubseteq \bot & (1{<}m{<}k{\le}n) \\ S_k \sqcap L_m \sqsubseteq \forall r.L_m \quad S_k \sqcap H_m \sqsubseteq \forall r.H_m & (1{<}k{<}m{\le}n) \\ L_k \sqcap \bigsqcap_{i<k} H_i \sqsubseteq S_k \quad S_k \sqsubseteq \forall r.H_k & (1 < k \le n) \end{array}$$

Next, the 2^n-r-chains are used to encode bit counters with 2^n positions. Those are 'concatenated' in ascending order to obtain a chain of double exponential length. We use the nominal o as origin of this chain and L_x, H_x as the new counter's low- and high-bit, respectively. The origin starts a 2^n-r-chain; low and high bit are mutually exclusive:

$$\{o\} \sqsubseteq Z \qquad H_x \sqcap L_x \sqsubseteq \bot$$

The Z_v-concept is set for o and propagated through the 2^n-r-chain setting all low-bits along the way.

$$\{o\} \sqsubseteq Z_v \qquad Z_v \sqsubseteq \forall r.Z_v \qquad Z_v \sqsubseteq L_x$$

We use the following technique to detect an 2^n-r-chain with all high-bits set: we initialize E_v to hold in every 2^n-r-chain origin and propagate it along the chain as long as all high-bits on the way are set. As soon as we encounter a low-bit, we propagate N_v instead.

$$Z \sqsubseteq E_v \quad E_v \sqcap H_x \sqsubseteq \forall r.E_v \quad L_x \sqsubseteq \forall r.N_v \quad E_v \sqcap N_v \sqsubseteq \bot$$

Now, if we arrive at the end of a 2^n-length counter with not all high bits set (detected by the fact that N_v or L_x hold), we establish a v-link to an individual that starts a new 2^n-length counter.

$$E \sqcap N_v \sqsubseteq \exists v Z \qquad E \sqcap L_x \sqsubseteq \exists v Z$$

Both r and v are made subroles of v_0 and we let roles $v_1, \ldots, v_n$ span v_0-chains of length $2^1, \ldots, 2^n$, respectively.

$$r \sqsubseteq v_0 \qquad v \sqsubseteq v_0 \qquad v_{i-1} \circ v_{i-1} \sqsubseteq v_i \quad (1 < i \leq n)$$

This establishes v_n-links between corresponding positions of two subsequent 2^n-length counters which then can be used to implement the step-wise increment.We use auxiliary concepts F_x and U_x to indicate whether the bit should be flipped in the subsequent counter.

The first bit of a counter is always flipped. If a high-bit is to be flipped then the bit on the next position has to be flipped as well. In case of a low bit, the next position bit (and all those on subsequent positions) will remain unflipped.

$$Z \sqsubseteq F_x \quad H_x \sqcap F_x \sqsubseteq \forall r.F_x \quad L_x \sqsubseteq \forall r.U_x \quad U_x \sqsubseteq \forall r.U_x$$

Next we implement the flipping mechanism.

$$\begin{array}{cc} H_x \sqcap F_x \sqsubseteq \forall v_n.L_x & L_x \sqcap F_x \sqsubseteq \forall v_n.H_x \\ H_x \sqcap U_x \sqsubseteq \forall v_n.H_x & L_x \sqcap U_x \sqsubseteq \forall v_n.L_x \end{array}$$

For establishing a grid, we first duplicate the axioms to create an independent counter (progressing along the h-role) for the y-direction of the grid. Thereby we reuse the 2^n-r-chains to carry the bit representations of the y coordinate.

$$\begin{array}{c} H_y \sqcap L_y \sqsubseteq \bot \quad \{o\} \sqsubseteq Z_h \quad Z_h \sqsubseteq \forall r.Z_h \quad Z_h \sqsubseteq L_y \\ Z \sqsubseteq E_h \quad E_h \sqcap H_y \sqsubseteq \forall r.E_h \quad L_y \sqsubseteq \forall r.N_h \quad E_h \sqcap N_h \sqsubseteq \bot \\ E \sqcap N_h \sqsubseteq \exists h.Z \qquad E \sqcap L_y \sqsubseteq \exists h.Z \\ r \sqsubseteq h_0 \quad h \sqsubseteq h_0 \quad h_{i-1} \circ h_{i-1} \sqsubseteq h_i \quad (1 < i \leq n) \\ Z \sqsubseteq F_y \quad H_y \sqcap F_y \sqsubseteq \forall r.F_y \quad L_y \sqsubseteq \forall r.U_y \quad U_y \sqsubseteq \forall r.U_y \\ H_y \sqcap F_y \sqsubseteq \forall h_n.L_y \qquad L_y \sqcap F_y \sqsubseteq \forall h_n.H_y \\ H_y \sqcap U_y \sqsubseteq \forall h_n.H_y \qquad L_y \sqcap U_y \sqsubseteq \forall h_n.L_y \end{array}$$

Now we make sure that x-counters don't change along h_n and y-counters don't change along v_n:

$$H_x \sqsubseteq \forall h_n.H_x \quad L_x \sqsubseteq \forall h_n.L_x \quad H_y \sqsubseteq \forall v_n.H_y \quad L_y \sqsubseteq \forall v_n.L_y$$

So far, the axioms essentially enforce a binary tree of double exponential depth, the leaves of which all have the same x-and y-coordinates. We now use an 'end' nominal to tie together all those leaves (based on their coordinates).

$$E \sqcap E_v \sqcap H_x \sqcap E_h \sqcap H_y \sqsubseteq \{e\}$$

All coordinate-equal individuals are forced to be identified by declaring the roles r, v and h inverse functional.

$$\top \sqsubseteq \leqslant 1\, r^-.\top \qquad \top \sqsubseteq \leqslant 1\, v^-.\top \qquad \top \sqsubseteq \leqslant 1\, h^-.\top$$

This finally enforces every model of the described knowledge base to contain a $2^{2^n} \times 2^{2^n}$ grid w.r.t. the vertical role v_n and the horizontal role h_n.

Encoding a Deterministic Turing Machine A *(deterministic) Turing machine* (TM) $\mathcal{M}$ is a tuple $(Q, \Sigma, \delta, q_0, Q_f)$ where Q is a finite set of *states*, Σ is a finite *alphabet* that includes a *blank* symbol $\Box$, $\delta : Q \times \Sigma \to Q \times \Sigma \times \{l, r\}$ is the *transition function*, $q_0 \in Q$ is the *initial state*, and $Q_f \subseteq Q$ is the set of *final states*. A *configuration* of $\mathcal{M}$ is a word $\alpha \in \Sigma^* Q \Sigma^*$. A configuration α' is the *successor* of a configuration α if one of the following holds:

1. $\alpha = w_l q \sigma \sigma_r w_r$, $\alpha' = w_l \sigma' q' \sigma_r w_r$, $\delta(q, \sigma) = (q', \sigma', r)$,
2. $\alpha = w_l q \sigma$, $\alpha' = w_l \sigma' q' \Box$, $\delta(q, \sigma) = (q', \sigma', r)$, or
3. $\alpha = w_l \sigma_l q \sigma w_r$, $\alpha' = w_l q' \sigma_l \sigma' w_r$, $\delta(q, \sigma) = (q', \sigma', l)$,

where $q \in Q$, $\sigma, \sigma', \sigma_l, \sigma_r \in \Sigma$, and $w_l, w_r \in \Sigma^*$.

Given a Turing machine $\mathcal{M}$ and a word $w \in \Sigma^*$, the *word problem* is to decide whether $\mathcal{M}$ *accepts* w, that is, whether there exists a sequence of successor configurations $\alpha_0, \ldots, \alpha_m$ such that $\alpha_0 = q_0 w$ and $\alpha_m = w_1 q_f w_2$ for some $q_f \in Q_f$.

For showing a 2-ExpTime lower bound, we assume that there exists a polynomial p such that for every w, m is bounded by $2^{2^{p(k)}}$, where $k = |Q| + |w|$. Note that we only need to represent configurations α with $|\alpha| < 2^{2^{p(k)}}$, as no larger configurations can occur during a run of length $2^{2^{p(k)}}$. We use a grid of size $2^{2^{p(k)}} \times 2^{2^{p(k)}}$, axiomatized as above, to represent the run of $\mathcal{M}$ on w. Each row along the h-axis represents a configuration at a time instant, and configurations evolve along the v-axis.

We introduce a concept name A_σ for every $\sigma \in \Sigma \cup \{\Box\}$ to indicate what symbol is written on a specific tape position. Moreover, we introduce concept names H_q for all $q \in Q$ to indicate that the head of the Turing machine is at a specific position in state q. The concept names $\overline{H_r}$ and $\overline{H_l}$ are used to indicate the absence of the head at a specific position. All A_σ have to be disjoint, likewise the head status concepts.

$$\begin{array}{cc} A_\sigma \sqcap A_{\sigma'} \sqsubseteq \bot \quad \sigma \neq \sigma' & H_q \sqcap H_{q'} \sqsubseteq \bot \quad q \neq q' \\ \overline{H_r} \sqcap H_q \sqsubseteq \bot & \overline{H_l} \sqcap H_q \sqsubseteq \bot \end{array}$$

Let $w = \sigma_0 \ldots \sigma_l$. We introduce concept names $I_0, \ldots I_{l+1}$ to realize the initial tape content.

$$\begin{array}{l} \{o\} \sqsubseteq I_0 \sqcap H_{q_0} \quad I_j \sqsubseteq A_{\sigma_j} \sqcap \forall h_n.I_{j+1} \quad (0 \leq j \leq l) \\ I_j \sqsubseteq \sqcap \overline{H_r} \quad \text{for} \quad 1 \leq j \leq l \\ I_{l+1} \sqsubseteq A_\Box \qquad I_{l+1} \sqsubseteq \forall h_n.I_{l+1} \end{array}$$

Then we implement the way a configuration determines its successor configuration. The following axioms have to be added for all $q \in Q$ and $\sigma \in \Sigma$:

$$\begin{array}{c} H_q \sqcap A_\sigma \sqsubseteq \forall v_n.A_{\sigma'} \sqcap \forall h_n.H_{q'} \quad \text{if } \delta(q, \sigma) = (q', \sigma', r) \\ H_q \sqcap A_\sigma \sqsubseteq \forall v_n.A_{\sigma'} \sqcap \forall h_n^-.H_{q'} \quad \text{if } \delta(q, \sigma) = (q', \sigma', l) \\ \overline{H_r} \sqcap A_\sigma \sqsubseteq \forall v_n.A_\sigma \qquad \overline{H_l} \sqcap A_\sigma \sqsubseteq \forall v_n.A_\sigma \\ H_q \sqsubseteq \forall h_n.\overline{H_r} \sqcap \forall h_n^-.\overline{H_l} \quad \overline{H_r} \sqsubseteq \forall h_n.\overline{H_r} \quad \overline{H_l} \sqsubseteq \forall h_n^-.\overline{H_l} \end{array}$$

As every model represents the deterministic run of $\mathcal{M}$ with input w, if $\mathcal{M}$ accepts w, then $A_{q_f}^{\mathcal{I}}$ must be nonempty for at least one $q_f \in Q_f$. By adding $A_{q_f} \sqsubseteq \bot$ for all $q_f \in Q_f$, we obtain a knowledge base that is unsatisfiable iff $\mathcal{M}$ is accepts w. We remark that the construction is polynomial, and that all axioms are in Horn-$\mathcal{SROIQ}$.