Rewriting the Description Logic $\mathcal{ALCHIQ}$ to Disjunctive Existential Rules: Extended Technical Report*

David Carral and **Markus Krötzsch**
Knowledge-Based Systems Group, TU Dresden
firstname.lastname@tu-dresden.de

Abstract

Especially in data-intensive settings, a promising reasoning approach for description logics (DLs) is to rewrite DL theories into sets of rules. Although many such approaches have been considered in the literature, there are still various relevant DLs for which no small rewriting (of polynomial size) is known. We therefore develop small rewritings for the DL $\mathcal{ALCHIQ}$ – featuring disjunction, number restrictions, and inverse roles – to disjunctive Datalog. By admitting existential quantifiers in rule heads, we can improve this result to yield only rules of bounded size, a property that is common to all rewritings that were implemented in practice so far.

1 Introduction

Among the many approaches towards efficient reasoning in description logics (DLs), consequence-preserving translations of DL theories into rule languages are among the most prominent. Table 1 gives an overview of works in this area. The natural strength of rule reasoners is their good scalability towards large sets of facts. Indeed, all of the rewritings in Table 1 are independent of the given facts (ABox), which can be used unchanged for reasoning with the rule-based theory. In general, we can describe this idea of rewriting as follows:

Definition 1. *Consider fragments $\mathcal{L}_1$ and $\mathcal{L}_2$ of first-order logic that include ground facts.*

An $\mathcal{L}_2$-theory $\mathcal{T}_2$ is a fact-entailment preserving rewriting *(or simply* rewriting *in this paper) of an $\mathcal{L}_1$-theory $\mathcal{T}_1$ if, for every set $\mathcal{F}$ of ground facts and every ground fact φ over the signature of $\mathcal{T}_1$, we have $\mathcal{T}_1, \mathcal{F} \models \varphi$ iff $\mathcal{T}_2, \mathcal{F} \models \varphi$.*

If such a rewriting can always be computed, then $\mathcal{L}_1$ is (effectively, fact-entailment preserving) rewritable *to $\mathcal{L}_2$.*

All of the works in Table 1 establish rewritability in this sense for the given fragments, and in particular preserve entailments of all role (i.e., binary) atoms. If only entailments of class (i.e., unary) atoms are of interest, then all uses of $\mathcal{ALC}$ in the table can also be replaced by $\mathcal{S}$ – and by $\mathcal{SR}$ at the cost of exponentially larger rewritings – based on common preprocessing techniques [Ortiz *et al.*, 2010]. Entailment of (complex) role atoms is also a special case of regular path query answering, which can be solved by combined methods that we do not discuss here [Simancik, 2012].

Work	Source	Target	Size
Hustadt *et al.* [2007]	$\mathcal{ALCHIQ}$	Datalog$^\vee$	exp. †
Eiter *et al.* [2012]	Horn-$\mathcal{SHIQ}$	Datalog	exp. †
Rudolph *et al.* [2012]	$\mathcal{SHIQ}b_s$	Datalog$^\vee$	exp. †
Bienvenu *et al.* [2014]	$\mathcal{SHI}$	Datalog$^\vee$	exp. †
Carral *et al.* [2018]	Horn-$\mathcal{ALCHOIQ}$	Datalog	exp. †
Carral *et al.* [2019b]	Horn-$\mathcal{SHIQ}$	Datalog	exp. †
	Horn-$\mathcal{SRIQ}$	Datalog	2exp.†
Ortiz *et al.* [2010]	Horn-$\mathcal{ALCHOIQ}$	Datalog	poly.
Ahmetaj *et al.* [2016]	$\mathcal{ALCHIO}$	Datalog$^\vee$	poly.
Krötzsch [2011]	$\mathcal{EL}^{++}$	Datalog	poly. †
Carral *et al.* [2019a]	Horn-$\mathcal{ALC}$	Datalog$^\exists$	poly. †

†: rules of bounded size that does not depend on input

Table 1: Rewritings of DL-type logics to rule languages, where Datalog$^\vee$ and Datalog$^\exists$ denote Datalog extended with disjunctions and existential quantifiers, respectively

While the works in Table 1 rely on diverse rewriting methods, they must all obey some complexity-related constraints. In particular, if $\mathcal{L}_1$ is rewritable to $\mathcal{L}_2$, then $\mathcal{L}_2$'s data complexity for fact entailment subsumes that of $\mathcal{L}_1$. Hence, Horn-DLs (P-complete in data) are rewritable to Datalog, while non-Horn DLs (co-NP-complete in data) supposedly are not.

Combined complexity is also a limiting factor. It is ExpTime-complete for Datalog and co-NExpTime-complete for Datalog$^\vee$, but drops to P and NP, respectively, if rule sizes are bounded († in Table 1). This explains why †-rewritings from (N)ExpTime-complete DLs to Datalog$^{(\vee)}$ must produce exponentially large rule sets. The alternative is to allow polynomially growing rule sizes (and especially predicate arities). The last two lines in Table 1 are rewritings that use constant rule sets and rewrite TBoxes to facts in (sub)polynomial time. This works for $\mathcal{EL}^{++}$ due to its P-complexity, and for Horn-$\mathcal{ALC}$ due to the high expressive power of Datalog$^\exists$ theories, even when these are constant.

Given this rich body of research in rule rewritings, there is

*Results from this technical report are published in an eponymous IJCAI 2020 paper, which should be used in citations [Carral and Krötzsch, 2020].

a surprising shortage of rewriting-based reasoners. KAON2 uses the exponential $\mathcal{SHIQ}$ rewriting of Hustadt *et al.* [2007], and DReW the polynomial $\mathcal{EL}^{++}$ rewriting of Krötzsch [2011] – both systems are discontinued. Rule reasoners, in contrast, have been thriving in recent years, and scalable systems exist both for Datalog$^\vee$ (e.g., answer set programming engines [Lifschitz, 2019]) and for Datalog$^\exists$ (e.g., engines for tuple-generating dependencies [Bellomarini *et al.*, 2018; Motik *et al.*, 2014; Urbani *et al.*, 2018]).

A possible explanation is that known rewritings still suffer from many shortcomings. Indeed, exponentially large rule sets (Table 1, lines 1–7) and rule sizes in the order of the ontology (lines 8–9) both impair practical performance, whereas static rewritings (lines 10–11) are often better implemented in dedicated "consequence-based" reasoners rather than relying on general-purpose rule reasoners [Kazakov *et al.*, 2013]. Moreover, Table 1 highlights that many DL feature combinations are not supported by any polynomial rewriting.

In this paper, we therefore study polynomial rewritings for hitherto unsupported DLs, in particular for the combination of number restrictions ($\mathcal{Q}$) and inverses ($\mathcal{I}$) on a non-Horn DL. The result are two new rewriting approaches for the DL $\mathcal{ALCHIQ}$: a rewriting to Datalog$^\vee$ that (unavoidably) requires unbounded (but still linear) rule sizes, and a rewriting to Datalog$^{\vee\exists}$ that achieves bounded rule sizes. Datalog with existential quantifiers is co-re-complete for fact entailment, but we show that our rewriting leads to rule sets for which entailment can be decided with standard algorithms, while still matching the original DL's co-NP data complexity. Both results are new, and – in the second case – also illustrate the potential advantage of considering rules with existential quantifiers as a target for rewriting.

Finally, we also consider Horn-$\mathcal{ALCHIQ}$, the disjunction-free fragment of $\mathcal{ALCHIQ}$. Whereas our rewriting of $\mathcal{ALCHIQ}$ to Datalog$^\vee$ and Datalog$^{\vee\exists}$, respectively, could be applied here, it produces rule sets that always contain disjunctions. We therefore combine several known results to obtain alternative, disjunction-free rewritings for this case.

This is an extended technical report with additional details on proofs that have been omitted from the conference version [Carral and Krötzsch, 2020].

2 Preliminaries

We introduce Datalog$^{\vee\exists}$ as a first-order rule language, and define Datalog, Datalog$^\vee$, and the DL $\mathcal{ALCHIQ}$ as fragments thereof. The *datalog-first disjunctive chase* provides a suitable deduction calculus for the Datalog$^{\vee\exists}$ fragments we rewrite to. We assume familiarity with standard notions of first-order logic, such as interpretations and homomorphisms.

2.1 Rules and Logic Fragments

We consider a first-order signature based on mutually disjoint, countably infinite sets of *constants* $\mathbf{C}$, *variables* $\mathbf{V}$, *nulls* $\mathbf{N}$, and *predicates* $\mathbf{P}$. The set of *terms* is $\mathbf{T} = \mathbf{C} \cup \mathbf{V} \cup \mathbf{N}$. A predicate $P \in \mathbf{P}$ has an arity $ar(P) \geq 1$, $\mathbf{P}^i = \{P \mid ar(P) = i\}$. We consider special predicates $\top, \bot \in \mathbf{P}^1$, and $\approx\, \in \mathbf{P}^2$, which are semantically interpreted as universal class, empty class, and equality. For

($\sqcap_{\mathsf{C}}$)	$A(x) \wedge B(x) \to C(x)$	$A \sqcap B \sqsubseteq C$
($\sqcup_{\mathsf{C}}$)	$A(x) \to B(x) \vee C(x)$	$A \sqsubseteq B \sqcup C$
($\forall$)	$A(x) \wedge R(x,y) \to B(y)$	$A \sqsubseteq \forall R.B$
($\exists$)	$A(x) \to \exists y.R(x,y) \wedge B(y)$	$A \sqsubseteq \exists R.B$
($\circ$)	$A(x) \wedge R(x,y) \wedge B(y) \to S(x,y)$	$A \circ R \circ B \sqsubseteq S$
($\leqslant$)	$R(x,y) \wedge R(x,z) \to y \approx z$	$\top \sqsubseteq \leqslant 1\, R.\top$
($\sqcap_{\mathsf{R}}$)	$R(x,y) \wedge S(x,y) \to V(x,y)$	$R \sqcap S \sqsubseteq V$
($\sqcup_{\mathsf{R}}$)	$R(x,y) \to S(x,y) \vee V(x,y)$	$R \sqsubseteq S \sqcup V$
($-$)	$R(y,x) \to S(x,y)$	$R^- \sqsubseteq S$

Figure 1: $\mathcal{ALCHIQ}$ rules ($A, B, C \in \mathbf{P}^1$ and $R, S, V \in \mathbf{P}^2 \backslash \{\approx\}$)

a logical expression or set thereof υ and a set of entities $\mathbf{E} \in \{\mathbf{C}, \mathbf{V}, \mathbf{N}, \mathbf{T}, \mathbf{P}\} \cup \{\mathbf{P}^i \mid i \geq 1\}$, let $\mathbf{E}_\upsilon$ be the set containing every element in $\mathbf{E}$ that occurs in υ. Lists of terms $t_1, \ldots, t_n$ are written as $\vec{t}$ and treated as sets when order is irrelevant. Letters x, y, z, w, and v, possibly with subscripts, always denote variables.

An *atom* is a formula $P(\vec{t})$ with $P \in \mathbf{P}^{|\vec{t}|}$ and $\vec{t} \subseteq \mathbf{T}$. A *fact* is a variable-free atom. A *disjunctive existential rule* is a null-free first-order logic formula of the form

$$\forall \vec{x}. \Big(\beta[\vec{x}] \to \bigvee\nolimits_{i=1}^{n} \exists \vec{y}_i . \eta_i[\vec{x}_i, \vec{y}_i] \Big). \qquad (\ddagger)$$

where $\beta[\vec{x}]$ and $\eta_i[\vec{x}_i, \vec{y}_i]$ are conjunctions of atoms using variables in the specified lists $\vec{x}_{(i)}$ and $\vec{y}_i$, which satisfy $\vec{x}_i \subseteq \vec{x}$ and $\vec{x} \cap \vec{y}_i = \emptyset$ for all $1 \leq i \leq n$. The rule has *body* $\beta[\vec{x}]$ and *head* $\bigvee_{i=1}^{n} \exists \vec{y}_i . \eta_i[\vec{x}_i, \vec{y}_i]$. We omit the universal quantifiers when writing rules. We use disjunctions in bodies as a shortcut, e.g., we write $\beta_1 \vee \beta_2 \to \exists \vec{y}.\eta$ instead of $\beta_1 \to \exists \vec{y}.\eta$ and $\beta_2 \to \exists \vec{y}.\eta$.

Definition 2. *Datalog$^{\vee\exists}$ is the language of all sets $\mathcal{R}$ of disjunctive existential rules. We consider some of its fragments:*

- Datalog$^\exists$*: Datalog$^{\vee\exists}$ without disjunction.*
- Datalog$^\vee$*: Datalog$^{\vee\exists}$ without existential quantifiers.*
- Datalog*: Datalog$^\vee$ without disjunctions.*
- *$\mathcal{ALCHIQ}$: rules of one of the forms given in Figure 1.*
- Horn-*$\mathcal{ALCHIQ}$: $\mathcal{ALCHIQ}$ without disjunction.*

Our definition of $\mathcal{ALCHIQ}$ is based on first-order representations of the axioms of the DL $\mathcal{SHIQ}b_s$ shown on the right of Figure 1. Arbitrary $\mathcal{ALCHIQ}$ axioms [Horrocks *et al.*, 2000] can be converted into these forms while preserving entailment of facts [Rudolph *et al.*, 2012, Section 6]. Note that this conversion can be computed in polynomial time if the number restrictions are encoded in unary.

An *ontology* $\mathcal{O}$ is a pair $\langle \mathcal{R}, \mathcal{F} \rangle$ of a rule set $\mathcal{R}$ and a null-free fact set $\mathcal{F}$. Without loss of generality, we assume $\mathbf{P}_\mathcal{F} \subseteq \mathbf{P}_\mathcal{R}$ and $\top, \bot, \approx\, \notin \mathbf{P}_\mathcal{F}$. We write $\mathcal{O} \models \upsilon$ if $\mathcal{O}$ entails a logical expression υ in first-order logic with $\approx$, $\top$, and $\bot$.

2.2 The Chase.

We introduce a general proof procedure for Datalog$^{\vee\exists}$ as the non-deterministic version of the *restricted chase* algorithm. Following Carral *et al.* [2019a], we prioritise Datalog$^\vee$ rules to increase the chance of termination.

A *substitution* is a partial function $\sigma : \mathbf{V} \to \mathbf{C} \cup \mathbf{N}$. For a logical expression υ, let $\upsilon\sigma$ be the result of replacing each variable x in υ by $\sigma(x)$ if the latter is defined. Given a fact set $\mathcal{F}$ and a rule ρ of the form (‡), σ is a *match* for ρ if $\beta\sigma \subseteq \mathcal{F}$ and σ is not defined on $\vec{y}_1 \cup \ldots \cup \vec{y}_n$.

Definition 3. *A* chase sequence *of an ontology $\langle \mathcal{R}, \mathcal{F} \rangle$ is a (finite or infinite) sequence of fact sets $\mathcal{F}_0, \mathcal{F}_1, \ldots$ such that*

1. *$\mathcal{F}_0 = \mathcal{F}$;*
2. *for every $\mathcal{F}_{i+1}$ with $i \geq 0$, there is a rule $\rho \in \mathcal{R}$ of the form (‡) and a match σ for ρ on $\mathcal{F}_i$ such that*
   - *(a) $\mathcal{F}_i \not\models \exists \vec{y}_i.\eta_i\sigma$ for all $i \in \{1, \ldots, n\}$,*
   - *(b) $\mathcal{F}_{i+1} = \mathcal{F}_i \cup \eta_j\hat{\sigma}$ for some $j \in \{1, \ldots, n\}$, where $\hat{\sigma}$ extends σ to $\vec{y}_j$ such that $\hat{\sigma}(y) \in \mathbf{N}$ is a fresh null for each $y \in \vec{y}_j$,*
   - *(c) if ρ contains an existential quantifier, then for all Datalog$^\vee$ rules $\rho' \in \mathcal{R}$ and all matches σ' of ρ' on $\mathcal{F}_i$, we have $\mathcal{F}_i \models \rho'\sigma'$ (*priority for Datalog$^\vee$*);*
3. *for every $\rho \in \mathcal{R}$ and substitution σ, there is some $k \geq 1$ such that $\mathcal{F}_i \models \rho\sigma$ for all $i \geq k$ (*fairness*).*

The set $\bigcup_{i \geq 0} \mathcal{F}_i$ is a chase *of $\langle \mathcal{R}, \mathcal{F} \rangle$.*

The chase process is non-deterministic in several ways. Disjunctions are handled by the choice of j in (2b). This affects which facts are derived. Another form of non-determinism is in the choice of rule application order, within the constraints of (2c) and (3). This does not affect which ground facts are eventually derived, but it may determine if the chase sequence is finite or not. A rule set $\mathcal{R}$ is *(universally) terminating* if, for all fact sets $\mathcal{F}$, all chase sequences of $\langle \mathcal{R}, \mathcal{F} \rangle$ are finite. The language of all terminating rules sets is called *terminating Datalog*$^{\vee\exists}$ (and similarly for other logics).

Definition 3 does not consider the semantics of the special predicates $\top$, $\bot$, and $\approx$, which must satisfy certain restrictions in all first-order interpretations: $\top$ and $\bot$ are interpreted as the domain and the empty set, respectively, and $\approx$ is used to denote equality. We treat these special predicates by axiomatisation. For a rule ρ, let $Ax(\rho)$ be the result of replacing all occurrences of $\top$, $\bot$, and $\approx$ by Top, Bot, and Eq, respectively. For a rule set $\mathcal{R}$, let $Ax(\mathcal{R})$ be the rule set that contains (i) $Ax(\rho)$ for all $\rho \in \mathcal{R}$, (ii) $\mathsf{Top}(c)$ for all $c \in \mathbf{C}_{\mathcal{R}}$, and (iii) the following rules instantiated for all $P \in \mathbf{P}_{\mathcal{R}} \setminus \{\top, \bot, \approx\}$ and $i \in \{1, \ldots, n\}$ where $n = ar(P)$:

$$P(x_1, \ldots, x_i, \ldots, x_n) \to \mathsf{Top}(x_i)$$
$$\mathsf{Bot}(y) \wedge \mathsf{Top}(x_1) \wedge \ldots \wedge \mathsf{Top}(x_n) \to P(x_1, \ldots, x_n)$$
$$P(x_1, \ldots, x_i, \ldots, x_n) \wedge \mathsf{Eq}(x_i, y) \to P(x_1, \ldots, y, \ldots, x_n)$$
$$\mathsf{Eq}(x, y) \to \mathsf{Eq}(y, x) \qquad \mathsf{Eq}(x, y) \wedge \mathsf{Eq}(y, z) \to \mathsf{Eq}(x, z)$$

The rules are clearly sound for the intended interpretation of the auxiliary predicates. The second rule type ensures that everything follows from a contradiction – practical systems would omit this for reasons of efficiency, and check for entailments over Bot as a special case. The final three rules are the usual equality theory, where we omit reflexivity since it does not lead to conclusions for any predicates other than Eq.

Fact 1. *An ontology $\mathcal{O} = \langle \mathcal{R}, \mathcal{F} \rangle$ entails a fact φ over a predicate in $\mathbf{P}_{\mathcal{O}} \setminus \{\top, \bot, \approx\}$ iff φ is in all chases of $\langle Ax(\mathcal{R}), \mathcal{F} \rangle$.*

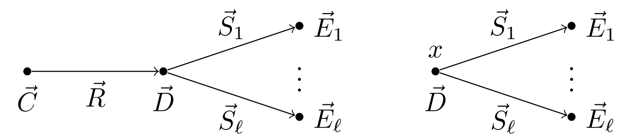

Figure 2: Sketch of the local environments as described by generic types (Type, left) and types for named elements x ($\mathsf{NamType}$, right)

3 $\mathcal{ALCHIQ}$ to Disjunctive Datalog

We will now specify a polynomial rewriting of $\mathcal{ALCHIQ}$ to Datalog$^\vee$, and establish its correctness. For the remainder of this section, let $\mathcal{R}$ be an arbitrary, fixed $\mathcal{ALCHIQ}$ rule set. Moreover, let $\mathcal{R}_\exists$ be the set of all rules of type ($\exists$) in $\mathcal{R}$; that is, the set of all non-Datalog rules in $\mathcal{R}$. We present a rewriting to establish the following main result.

Theorem 2. *$\mathcal{ALCHIQ}$ is poly-time rewritable to Datalog$^\vee$.*

Our rewriting is based on types that define (shapes of) local environments of domain elements in a DL interpretation. Sketches for the intuitive notions of "local environment" that we consider are shown in Figure 2. The types abstract from pairs of elements c and d that are directly related, by specifying the classes of c, the classes of d, and the set of (forward or inverse) roles relating them. To support $\mathcal{ALCHIQ}$, the type further considers, for each rule $\rho \in \mathcal{R}_\exists$, a successor w_ρ of d that witnesses the satisfaction of ρ on d. This successor is again described by the (forward or inverse) roles between d and w_ρ, and by the classes of w_ρ.

A type therefore is a tuple $\langle C, R, D, S_1, E_1, \ldots, S_\ell, E_\ell \rangle$ where $\ell = |\mathcal{R}_\exists|$, $C, D, E_i \subseteq \mathbf{P}^1$, and $R, S_i \subseteq \mathbf{P}^2 \setminus \{\approx\}$ for $i \in \{1, \ldots, \ell\}$. We encode types in Datalog$^\vee$ using a predicate Type of arity $(2+\ell)\cdot|\mathbf{P}^1_{\mathcal{R}} \cup \{\top, \bot\}| + (1+\ell)\cdot 2|\mathbf{P}^2_{\mathcal{R}} \setminus \{\approx\}|$, where each parameter can be one of the fresh individuals $\{0, 1\}$, indicating the presence or absence of an element in one of the sets that constitute the type. This approach follows Ortiz *et al.* [2010]. We use the following fixed lists of variables in Type atoms:

$$\vec{C} = c_{A_1}, \ldots, c_{A_n} \qquad \vec{R} = r_{R_1}, r_{R_1^-} \ldots, r_{R_m}, r_{R_m^-}$$
$$\vec{D} = d_{A_1}, \ldots, d_{A_n} \qquad \vec{W} = \vec{W}^{\rho_1}, \ldots, \vec{W}^{\rho_\ell}$$

for some fixed orderings $\mathbf{P}^1_{\mathcal{R}} \cup \{\top, \bot\} = \{A_1, \ldots, A_n\}$, $\mathbf{P}^2_{\mathcal{R}} \setminus \{\approx\} = \{R_1, \ldots, R_m\}$, and $\mathcal{R}_\exists = \{\rho_1, \ldots, \rho_\ell\}$; and with $\vec{W}^\rho = w^\rho_{R_1}, w^\rho_{R_1^-} \ldots, w^\rho_{R_m}, w^\rho_{R_m^-}, w^\rho_{A_1}, \ldots, w^\rho_{A_n}$ for every $\rho \in \mathcal{R}_\exists$. We will frequently encounter atoms $\mathsf{Type}(\vec{C}, \vec{R}, \vec{D}, \vec{W})$, which we further abbreviate as $\mathsf{Type}(\vec{V})$. We will further use facts of the form $\mathsf{NamType}(x, \vec{D}, \vec{W})$ to encode the type of a named individual x, specified by its classes ($\vec{D}$) and $\mathcal{R}_\exists$ witnesses ($\vec{W}$).

Definition 4. *Let $\mathsf{rew}^\vee(\mathcal{R})$ be the Datalog$^\vee$ rule set that contains (i) $Ax(\mathcal{R} \setminus \mathcal{R}_\exists)$; (ii) the rule set in Figure 3; and (iii), for every rule $\rho \in \mathcal{R}$, the corresponding rules in Figure 4.*

In this rewriting, the rules of (i) are (axiomatised) first-order versions of all non-existential ontology axioms. However, these rules will only apply to named individuals, since models of the rewritten Datalog$^\vee$ rules do not contain domain

$$\mathsf{Top}(x) \to \bigvee_{\vec{c}\in\{0,1\}^{|\vec{D}|+|\vec{W}|}} \big(\mathsf{NamType}(x,\vec{c}) \wedge \mathsf{Type}(0,\ldots,0,\vec{c})\big) \tag{1}$$

$$\mathsf{Type}(\vec{C},\vec{R},\vec{D},\vec{W}^{\rho_1},\ldots,\vec{W}^{\rho_\ell}) \to \bigvee_{\vec{c}\in\{0,1\}^{|\vec{W}|}} \mathsf{Type}(\vec{D},\vec{W}^{\rho_i},\vec{c}) \qquad i\in\{1,\ldots,\ell\} \tag{2}$$

$$\mathsf{NamType}(x,\vec{D},\vec{W})[d_A/0] \wedge A(x) \to \mathsf{Bot}(0) \qquad A \in \mathbf{P}^1_{\mathcal{R}} \setminus \{\top,\bot\} \tag{3}$$

$$\mathsf{Eq}(0,1) \vee \mathsf{Type}(\vec{C},\vec{R},\vec{D},\vec{W})[d_\bot/1] \to \mathsf{Bot}(0) \tag{4}$$

$$\mathsf{Type}(\vec{C},\vec{R},\vec{D},\vec{W})[d_\top/0, v/1] \to \mathsf{Bot}(0) \qquad v \in \vec{R}\cup\vec{D} \tag{5}$$

Figure 3: Part of the Datalog$^\vee$ rewriting $\mathsf{rew}^\vee(\mathcal{R})$ of $\mathcal{ALCHIQ}$ rule set $\mathcal{R}$; $[x_1/a_1,\ldots,x_n/a_n]$ is the substitution that maps every x_i to a_i

$$\begin{array}{llll}
(\ulcorner\mathsf{C}\urcorner) & \mathsf{Type}(\vec{V})[d_A/1, d_B/1, d_C/0] \to \mathsf{Bot}(0) & (\llcorner\mathsf{C}\lrcorner) & \mathsf{Type}(\vec{V})[d_A/1, d_B/0, d_C/0] \to \mathsf{Bot}(0) \\
(\forall) & \mathsf{Type}(\vec{V})[c_A/1, r_R/1, d_B/0] \to \mathsf{Bot}(0) & (\exists) & \mathsf{Type}(\vec{V})[d_A/1, w^\rho_R/0] \to \mathsf{Bot}(0) \\
 & \mathsf{Type}(\vec{V})[c_B/0, r_{R^-}/1, d_A/1] \to \mathsf{Bot}(0) & & \mathsf{Type}(\vec{V})[d_A/1, w^\rho_B/0] \to \mathsf{Bot}(0) \\
(\circ) & \mathsf{Type}(\vec{V})[c_A/1, r_R/1, d_B/1, r_S/0] \to \mathsf{Bot}(0) & (\ulcorner\mathsf{R}\urcorner) & \mathsf{Type}(\vec{V})[r_R/1, r_S/1, r_V/0] \to \mathsf{Bot}(0) \\
 & \mathsf{Type}(\vec{V})[c_B/1, r_{R^-}/1, d_A/1, r_{S^-}/0] \to \mathsf{Bot}(0) & & \mathsf{Type}(\vec{V})[r_{R^-}/1, r_{S^-}/1, r_{V^-}/0] \to \mathsf{Bot}(0) \\
(\llcorner\mathsf{R}\lrcorner) & \mathsf{Type}(\vec{V})[r_R/1, r_S/0, r_V/0] \to \mathsf{Bot}(0) & (-) & \mathsf{Type}(\vec{V})[r_{R^-}/1, r_S/0] \to \mathsf{Bot}(0) \\
 & \mathsf{Type}(\vec{V})[r_{R^-}/1, r_{S^-}/0, r_{V^-}/0] \to \mathsf{Bot}(0) & & \mathsf{Type}(\vec{V})[r_R/1, r_{S^-}/0] \to \mathsf{Bot}(0)
\end{array}$$

$$\begin{aligned}
(\leqslant)\ \{&\mathsf{Type}(\vec{V})[w^{\rho'}_R/1, w^{\rho''}_R/1] \to \mathsf{Eq}(w^{\rho'}_S, w^{\rho''}_S) \wedge \mathsf{Eq}(w^{\rho'}_{S^-}, w^{\rho''}_{S^-}) \wedge \mathsf{Eq}(w^{\rho'}_A, w^{\rho''}_A),\\
&\mathsf{Type}(\vec{V})[r_{R^-}/1, w^{\rho'}_R/1] \to \mathsf{Eq}(r_S, w^{\rho'}_{S^-}) \wedge \mathsf{Eq}(r_{S^-}, w^{\rho'}_S) \wedge \mathsf{Eq}(c_A, w^{\rho'}_A),\\
&R(x,y) \wedge \mathsf{NamType}(x,\vec{D},\vec{W})[w^{\rho'}_R/1, w^{\rho'}_S/1] \to S(x,y), R(x,y) \wedge \mathsf{NamType}(x,\vec{D},\vec{W})[w^{\rho'}_R/1, w^{\rho'}_{S^-}/1] \to S(y,x),\\
&R(x,y) \wedge \mathsf{NamType}(x,\vec{D},\vec{W})[w^{\rho'}_R/1, w^{\rho'}_A/1] \to A(y) \mid \rho',\rho'' \in \mathcal{R}_\exists, A \in \mathbf{P}^1_{\mathcal{R}} \setminus \{\top,\bot\}, S \in \mathbf{P}^2_{\mathcal{R}} \setminus \{\approx\}\}
\end{aligned}$$

Figure 4: Rewriting $\mathcal{ALCHIQ}$ rules ρ to Datalog$^\vee$; $[x_1/a_1,\ldots,x_n/a_n]$ is the substitution that maps every x_i to a_i

elements to represent the anonymous individuals that are relevant in DL models. Such anonymous elements instead are represented indirectly by recording their types, which is accomplished by rules (ii) and (iii). Rules (1) and (2) trigger the construction of types by requiring that named individuals have types (1), and that existing types have compatible successor types that satisfy existential restrictions (2). All other rules in Figures 3 and 4 then restrict these types by excluding contradicting cases. Rule (3) ensures the consistent assignment of classes to named individuals, rule (4) excludes some basic inconsistencies, and rule (5) ensures that the second element ("d") of any type must satisfy $\top$ if it satisfies anything at all. We do allow the case where all bits for $\vec{R}$ and $\vec{D}$ are 0, which occurs, e.g., for named individuals without successors.

Rules in Figure 4 encode the meaning of specific axioms when applied to the local structure of types as sketched in Figure 2. Most rules are straightforward, but the rules $(\leqslant)$ for expressing functionality require some explanation. Note that these rules are instantiated for various combinations of existential restrictions, class names, and role names. The rules in the first line encode equality of two existential successors (illustrated by elements in classes $\vec{E}_i$ in Figure 2); the rules in the second line encode equality of an existential successor and the first element of the type (illustrated by the element in classes $\vec{C}$ in Figure 2). Using Eq in these lines ensures that bits that encode the equated elements' type data are identical, since equality of 0 and 1 is disallowed by (4). The rules in the last two lines capture cases where predicates of named individuals can be derived from predicates of successor elements (represented by bits in a type). Note that we cannot use Eq for achieving the latter. Equality of several named individuals is already supported by the corresponding axiom in $Ax(\mathcal{R}\setminus\mathcal{R}_\exists)$.

For better clarity, we have chosen a presentation of $\mathsf{rew}^\vee(\mathcal{R})$ that is not polynomial, since rules (1) and (2) are of exponential size. To define a polynomial rewriting, we can simply substitute rule (1) with the rules

$$\begin{aligned}
\mathsf{Type}(\vec{V}) &\to \mathsf{Type}_1(\vec{D},\vec{W}^\rho,0) \vee \mathsf{Type}_1(\vec{D},\vec{W}^\rho,1)\\
\mathsf{Type}_i(\vec{D},\vec{W}^\rho,\vec{x}_i) &\to \bigvee_{j\in\{0,1\}} \mathsf{Type}_{i+1}(\vec{D},\vec{W}^\rho,\vec{x}_i,j)\\
\mathsf{Type}_k(\vec{V}) &\to \mathsf{Type}(\vec{V})
\end{aligned}$$

where $k = |\vec{V}|$, the second rule is instantiated for all $i = 1,\ldots,k-1$, and $\vec{x}_i$ is a list of variables of size i. We can replace the rules of type (2) in an analogous manner.

To establish Theorem 2, it remains to proof soundness

(Lemma 3) and completeness (Lemma 4).

Lemma 3. *For all fact sets $\mathcal{F}$ and all facts φ with a predicate in $\mathbf{P}_{\mathcal{R}} \setminus \{\top, \bot, \approx\}$, if $\langle \mathcal{R}, \mathcal{F}\rangle \not\models \varphi$ then $\langle \mathsf{rew}^{\vee}(\mathcal{R}), \mathcal{F}\rangle \not\models \varphi$.*

Proof. Let $\mathcal{O} = \langle \mathcal{R}, \mathcal{F}\rangle$. Since $\mathcal{O} \not\models \varphi$ there is a model $\mathcal{M} \models \mathcal{O}$ with $\mathcal{M} \not\models \varphi$. We define a model $\mathcal{I}$ of $\mathsf{rew}^{\vee}(\mathcal{R}) \cup \mathcal{F}$ and $\mathcal{I} \not\models \varphi$ by specifying all (ground) facts that $\mathcal{I}$ satisfies:

$\mathcal{I} \models \mathsf{Type}(0, \ldots, 0), \mathsf{Named}(a), \mathsf{Top}(a)$ for all $a \in \mathbf{C}_{\mathcal{O}}$
$\mathcal{I} \models A(a)$ for all $A \in \mathbf{P}^1_{\mathcal{R}} \setminus \{\top, \bot\}$ with $a^{\mathcal{M}} \in A^{\mathcal{M}}$
$\mathcal{I} \models R(a, b)$ for all $R \in \mathbf{P}^2_{\mathcal{R}}$ with $\langle a^{\mathcal{M}}, b^{\mathcal{M}}\rangle \in R^{\mathcal{M}}$
$\mathcal{I} \models \mathsf{Eq}(a, b)$ if $a^{\mathcal{M}} = b^{\mathcal{M}}$

For specifying $\mathsf{NamType}$ and Type, we first introduce some notation. For every $\rho \in \mathcal{R}_{\exists}$ of form $\rho = A(x) \to \exists y.R(x, y) \wedge B(y)$, and every $\delta \in A^{\mathcal{M}}$, there is an element $\gamma \in B^{\mathcal{M}}$ with $\langle \delta, \gamma\rangle \in R^{\mathcal{M}}$. Let $\mathit{Sat}^{\rho}_{\mathcal{M}}(\delta)$ denote one (fixed) such element.

Given elements $\delta, \gamma \in \Delta^{\mathcal{M}}$, we define lists of constants 0 and 1 to represent their type in $\mathcal{M}$ in the scheme of $\mathsf{Type}(\vec{C}, \vec{R}, \vec{D}, \vec{W})$. Let $[\![\delta]\!]_C = \vec{C}\sigma$ for $\sigma(c_A) = 1$ if $\delta \in A^{\mathcal{M}}$ and $\sigma(c_A) = 0$ if $\delta \notin A^{\mathcal{M}}$; $[\![\delta, \gamma]\!]_R = \vec{R}\sigma$ for $\sigma(r_R) = 1$ if $\langle \delta, \gamma\rangle \in R^{\mathcal{M}}$, $\sigma(r_R) = 0$ if $\langle \delta, \gamma\rangle \notin R^{\mathcal{M}}$, $\sigma(r_{R^-}) = 1$ if $\langle \gamma, \delta\rangle \in R^{\mathcal{M}}$, $\sigma(r_{R^-}) = 0$ if $\langle \gamma, \delta\rangle \notin R^{\mathcal{M}}$; and $[\![\delta, \gamma]\!]_{RD} = [\![\delta, \gamma]\!]_R, [\![\gamma]\!]_C$. Finally, let $[\![\delta]\!]_W$ be the list $\vec{u}^{\rho_1}, \ldots, \vec{u}^{\rho_\ell} \in \{0, 1\}$ for the rules $\{\rho_1, \ldots, \rho_\ell\} = \mathcal{R}_{\exists}$ as in the definition of $\vec{W}$, such that $\vec{u}^{\rho} = [\![\delta, \mathit{Sat}^{\rho}_{\mathcal{M}}(\delta)]\!]_{RD}$ if $\mathit{Sat}^{\rho}_{\mathcal{M}}(\delta)$ is defined, and $\vec{u}^{\rho} = 0^{|\vec{R}|+|\vec{D}|}$ otherwise.

For all $a \in \mathbf{C}_{\mathcal{O}}$, let $\mathcal{I} \models \mathsf{NamType}(a, [\![a^{\mathcal{M}}]\!]_C, [\![a^{\mathcal{M}}]\!]_W)$ and $\mathcal{I} \models \mathsf{Type}(0^{|\vec{C}|+|\vec{R}|}, [\![a^{\mathcal{M}}]\!]_C, [\![a^{\mathcal{M}}]\!]_W)$. Furthermore, for all $\delta \in \Delta^{\mathcal{M}}$ and $\rho \in \mathcal{R}_{\exists}$ such that $\mathit{Sat}^{\rho}_{\mathcal{M}}(\delta) = \gamma$, we set $\mathcal{I} \models \mathsf{Type}([\![\delta]\!]_C, [\![\delta, \gamma]\!]_{RD}, [\![\gamma]\!]_W)$.

$\mathcal{I} \models \mathit{Ax}(\mathcal{R} \setminus \mathcal{R}_{\exists})$ since all relevant predicates agree with $\mathcal{M}$ on named elements. $\mathcal{I}$ satisfies all rules in Figure 3, due to our coherent definition of types for named individuals, and since $\mathcal{M}$ satisfies $\mathcal{O}$. Satisfaction of rules in Figure 4 is easy to check for most rules, e.g., for $A \sqcap B \sqsubseteq C \in \mathcal{R}$, rule $\mathsf{Type}(\vec{V})[d_A/1, d_B/1, d_C/0] \to \mathsf{Bot}(0)$ is satisfied since the bits for $\vec{D}$ always have the form $[\![\delta]\!]_C$ for some $\delta \in \Delta^{\mathcal{M}}$. □

Lemma 4. *For all fact sets $\mathcal{F}$ and all facts φ with a predicate in $\mathbf{P}_{\mathcal{R}} \setminus \{\top, \bot, \approx\}$, if $\langle \mathsf{rew}^{\vee}(\mathcal{R}), \mathcal{F}\rangle \not\models \varphi$ then $\langle \mathcal{R}, \mathcal{F}\rangle \not\models \varphi$.*

Proof. Since $\langle \mathsf{rew}^{\vee}(\mathcal{R}), \mathcal{F}\rangle \not\models \varphi$, there is a model $\mathcal{I} \models \mathsf{rew}^{\vee}(\mathcal{R}) \cup \mathcal{F}$ with $\mathcal{I} \not\models \varphi$. To show $\langle \mathcal{R}, \mathcal{F}\rangle \not\models \varphi$, we define a model $\mathcal{M} \models \mathcal{R} \cup \mathcal{F}$ with $\mathcal{M} \not\models \varphi$ through an iterative construction.

Initially, let $\mathcal{M} = \mathcal{M}_0$ with $\Delta^{\mathcal{M}_0} = \{[a]_{\approx} \mid a \in \mathbf{C}_{\mathcal{R}}\}$ where $[a]_{\approx} = \{b \mid \mathcal{I} \models \mathsf{Eq}(a, b)\}$, and set $a^{\mathcal{M}_0} = [a]_{\approx}$ for all $a \in \mathbf{C}_{\mathcal{F}}$. The interpretations of other $c \in \mathbf{C} \setminus \mathbf{C}_{\mathcal{F}}$ is arbitrary. We further set $a^{\mathcal{M}_0} \in A^{\mathcal{M}_0}$ for $A \in \mathbf{P}^1_{\mathcal{R}}$ if $\mathcal{I} \models A(a)$, and $\langle a^{\mathcal{M}_0}, b^{\mathcal{M}_0}\rangle \in R^{\mathcal{M}_0}$ for $R \in \mathbf{P}^2_{\mathcal{R}} \setminus \{\approx\}$ if $\mathcal{I} \models R(a, b)$.

Then, we extend $\mathcal{M}$ by applying extension rules in a fair way. In each step, select an element $\delta \in \Delta^{\mathcal{M}}$ that has not been chosen before. If $\delta = a^{\mathcal{M}}$ for some $a \in \mathbf{C}_{\mathcal{F}}$, let σ be a substitution for which $\mathcal{I} \models \mathsf{NamType}(a, \vec{D}, \vec{W})\sigma$. Otherwise δ is of form $\delta = \gamma.\rho$ with $\delta \in \Delta^{\mathcal{M}}$ and $\rho \in \mathcal{R}_{\exists}$, and we let σ be a substitution for which $\mathcal{I} \models \mathsf{Type}(\vec{C}, \vec{R}, \vec{D}, \vec{W})\sigma$ with $\vec{C}\sigma = [\![\gamma]\!]_C$, $\vec{R} = [\![\gamma, \delta]\!]_R$, and $\vec{D}\sigma = [\![\delta]\!]_C$, where we use the notation from the proof of Lemma 3.

Then, for every $\rho \in \mathcal{R}_{\exists}$, we add some fresh $\delta.\rho \in \Delta^{\mathcal{M}}$, and extend $\mathcal{M}$ as follows: for all $A \in \mathbf{P}^1_{\mathcal{R}}$, let $\delta.\rho \in A^{\mathcal{M}}$ if $\sigma(w^{\rho}_A) = 1$, and, for all $R \in \mathbf{P}^2_{\mathcal{R}} \setminus \{\approx\}$, let $\langle \delta, \delta.\rho\rangle \in R^{\mathcal{M}}$ if $\sigma(w^{\rho}_R) = 1$ and $\langle \delta.\rho, \delta\rangle \in R^{\mathcal{M}}$ if $\sigma(w^{\rho}_{R^-}) = 1$. We say that $\delta.\rho$ is *unnecessary* if there is a homomorphism from $\mathcal{M}$ to the sub-interpretation obtained by removing $\delta.\rho$ from the domain; in this case, we immediately remove $\delta.\rho$ from $\mathcal{M}$. This completes the step performed for each element δ.

This iterative construction leads to a monotonically increasing sequence of interpretations $\mathcal{M}_0, \mathcal{M}_1, \ldots$ with union $\mathcal{M} = \bigcup_{i \geq 0} \mathcal{M}_i$. We can show by induction that each $\mathcal{M}_i$ satisfies all of the Datalog$^{\vee}$ rules in $\mathcal{R}$. The rules with existential quantifiers in $\mathcal{R}$ are satisfied by $\mathcal{M}$ because we apply the extension rule fairly to each domain element in $\Delta^{\mathcal{M}}$. □

4 $\mathcal{ALCHIQ}$ to Terminating Datalog$^{\vee\exists}$

We will now specify a polynomial rewriting of $\mathcal{ALCHIQ}$ to Datalog$^{\vee\exists}$. Our main result improves upon Theorem 2 by avoiding the need for predicates of unbounded arity:

Theorem 5. *$\mathcal{ALCHIQ}$ is poly-time rewritable into terminating Datalog$^{\vee\exists}$ rules of bounded size.*

For the remainder of this section, let $\mathcal{R}$ be an arbitrary, fixed $\mathcal{ALCHIQ}$ rule set. Moreover, let $\mathcal{R}_{\exists}$ and $\mathcal{R}_{\leqslant}$ be the sets of all rules of type ($\exists$) and ($\leqslant$) in $\mathcal{R}$, respectively. Theorem 5 is established by the following rewriting.

Definition 5. *Let $\mathsf{rew}^{\vee\exists}(\mathcal{R})$ be the Datalog$^{\vee\exists}$ rule set that contains the rules shown in Figure 5 and the rule set obtained from $\mathit{Ax}(\mathcal{R} \setminus (\mathcal{R}_{\exists} \cup \mathcal{R}_{\leqslant}))$ by replacing every predicate P with a fresh predicate $\mathbb{P}$ of the same arity.*

Note that Figure 5 uses predicates P and $\mathbb{P}$, as well as further variants $P^{\neg}$ and $\mathbb{P}^{\neg}$, which indicate the absence of a true atom. The fresh predicates (i.e., predicates such as $\mathbb{P}$) allow us to reserve regular ontology predicates for relations between named individuals, which we mark by Named. This relationship between the predicate names is expressed in the rules of (6). It is required since we often treat named individuals differently from individuals that were introduced to satisfy existential restrictions (see, e.g., rules of type (10)).

Intuitively, the rules implement a proof procedure that is similar to tableau calculi in DLs, where we construct a finite representation of a model by applying rules for each type of axiom iteratively. For example, rules of type (9) introduce a successor element to satisfy an existential restriction. Essential tableau-like structures are captured by predicates Named and $\mathsf{Unnamed}$, and by Succ (where $\mathsf{Succ}(x, y)$ means that y was created to satisfy an existential restriction on x). Facts $\mathsf{SmCl}_{\mathbb{A}_i}(x, y)$ mean that x and y agree on all classes $\mathbb{A}_1, \ldots, \mathbb{A}_i$ for a fixed ordering $A_1, \ldots, A_n \in \mathbf{P}^1_{\mathcal{R}} \setminus \{\top, \bot\}$. Facts $\mathsf{SmRl}_{\mathbb{R}_i}(x, y, z, w)$ are analogous for roles between class-equivalent pairs $\langle x, y\rangle$ and $\langle z, w\rangle$. The "equivalence" of elements y and w ($\mathsf{SameTyp}(y, w)$) is inferred by (17).

To ensure that the resulting structure is finite, however, we sometimes need to reuse anonymous elements. Tableau procedures accomplish this by a method known as *blocking*, and our approach is similar to *pairwise anywhere blocking* [Motik

$$\{P(\vec{x}) \to \mathbb{P}(\vec{x}) \wedge \bigwedge_{x\in\vec{x}} \mathsf{Named}(x), \mathbb{P}(\vec{x}) \wedge \bigwedge_{x\in\vec{x}} \mathsf{Named}(x) \to P(\vec{x}) \mid P \in \mathbf{P}_\mathcal{R} \cup \{\mathsf{Top}, \mathsf{Bot}, \mathsf{Eq}\} \setminus \{\top, \bot, \approx\}\} \cup \tag{6}$$

$$\{\mathsf{Unnamed}(x) \to \mathbb{A}(x) \vee \mathbb{A}^\neg(x), \mathbb{A}(x) \wedge \mathbb{A}^\neg(x) \to \mathsf{Bot}(x) \mid A \in \mathbf{P}^1_\mathcal{R} \setminus \{\top, \bot\}\} \cup \tag{7}$$

$$\{\mathsf{Succ}(x,y) \to \big(\mathbb{R}(x,y) \vee \mathbb{R}^\neg(x,y)\big) \wedge \big(\mathbb{R}(y,x) \vee \mathbb{R}^\neg(y,x)\big), \mathbb{R}(x,y) \wedge \mathbb{R}^\neg(x,y) \to \mathsf{Bot}(x) \mid R \in \mathbf{P}^2_\mathcal{R} \setminus \{\approx\}\} \cup \tag{8}$$

$$\{\mathbb{A}(x) \to \exists y.\mathbb{R}(x,y) \wedge \mathbb{B}(y) \wedge \mathsf{Succ}(x,y) \wedge \mathsf{Unnamed}(y) \wedge \mathsf{Top}(y) \mid A(x) \to \exists y.R(x,y) \wedge B(y) \in \mathcal{R}_\exists\} \cup \tag{9}$$

$$\begin{aligned}&\{\mathbb{R}(x,y) \wedge \mathbb{R}(x,z) \wedge \mathsf{Succ}(x,y) \wedge \big(\mathsf{Succ}(x,z) \vee \mathsf{Succ}(z,x) \vee \mathsf{Named}(x)\big) \to \mathsf{Copy}(x,y,z),\\ &\ \mathbb{R}(x,y) \wedge \mathbb{R}(x,z) \wedge \mathsf{Named}(x) \wedge \mathsf{Named}(y) \wedge \mathsf{Named}(z) \to \mathsf{Eq}(y,z) \mid R(x,y) \wedge R(x,z) \to y \approx z \in \mathcal{R}_\leq\} \cup\end{aligned} \tag{10}$$

$$\{\mathsf{Copy}(x,y,z) \wedge \mathbb{A}(y) \to \mathbb{A}(z) \mid A \in \mathbf{P}^1_\mathcal{R} \setminus \{\top, \bot\}\} \cup \tag{11}$$

$$\{\mathbb{R}(x,y) \wedge \mathsf{Copy}(x,y,z) \to \mathbb{R}(x,z), \mathbb{R}(y,x) \wedge \mathsf{Copy}(x,y,z) \to \mathbb{R}(z,x) \mid R \in \mathbf{P}^2_\mathcal{R} \setminus \{\approx\}\} \cup \tag{12}$$

$$\{\big(\mathbb{A}_1(x) \wedge \mathbb{A}_1(z)\big) \vee \big(\mathbb{A}_1^\neg(x) \wedge \mathbb{A}_1^\neg(z)\big) \to \mathsf{SmCl}_{\mathbb{A}_1}(x,z)\} \cup \tag{13}$$

$$\{\mathsf{SmCl}_{\mathbb{A}_{i-1}}(x,z) \wedge \big(\big(\mathbb{A}_i(x) \wedge \mathbb{A}_i(z)\big) \vee \big(\mathbb{A}_i^\neg(x) \wedge \mathbb{A}_i^\neg(z)\big)\big) \to \mathsf{SmCl}_{\mathbb{A}_i}(x,z) \mid 2 \leq i \leq n\} \cup \tag{14}$$

$$\{\mathsf{SmCl}_{\mathbb{A}_n}(x,z) \wedge \mathsf{SmCl}_{\mathbb{A}_n}(y,w) \wedge \big(\mathbb{R}_1(x,y) \wedge \mathbb{R}_1(z,w)\big) \vee \big(\mathbb{R}_1^\neg(x,y) \wedge \mathbb{R}_1^\neg(z,w)\big) \to \mathsf{SmRl}_{\mathbb{R}_1}(x,y,z,w)\} \cup \tag{15}$$

$$\{\mathsf{SmRl}_{\mathbb{R}_{i-1}}(x,y,z,w) \wedge \big(\big(\mathbb{R}_i(x,y) \wedge \mathbb{R}_i(z,w)\big) \vee \big(\mathbb{R}_i^\neg(x,y) \wedge \mathbb{R}_i^\neg(z,w)\big)\big) \to \mathsf{SmRl}_{\mathbb{R}_i}(x,y,z,w) \mid 2 \leq i \leq m\} \cup \tag{16}$$

$$\{\mathsf{Succ}(x,y) \wedge \mathsf{Succ}(z,w) \wedge \mathsf{SmRl}_{\mathbb{R}_m}(x,y,z,w) \wedge \mathsf{SmRl}_{\mathbb{R}_m}(y,x,w,z) \to \mathsf{SameTyp}(y,w)\} \cup \tag{17}$$

$$\begin{aligned}&\{\mathsf{SameTyp}(x,z) \wedge \mathsf{Succ}(x,y) \wedge \mathbb{P}(x,y) \to \mathsf{Succ}(z,y) \wedge \mathbb{P}(z,y),\\ &\ \mathsf{SameTyp}(x,z) \wedge \mathsf{Succ}(x,y) \wedge \mathbb{P}(y,x) \to \mathsf{Succ}(x,y) \wedge \mathbb{P}(y,z) \mid P \in (\mathbf{P}^2_\mathcal{R} \cup \{R^\neg \mid R \in \mathbf{P}^2_\mathcal{R}\}) \setminus \{\approx, \approx^\neg\}\}\end{aligned} \tag{18}$$

Figure 5: Mapping the $\mathcal{ALCHIQ}$ Rule Set $\mathcal{R}$ into the Terminating Datalog$^{\vee\exists}$ Rule Set $rw_4(\mathcal{R})$ ($A_1, \ldots, A_n$ and $R_1, \ldots, R_m$ are lists without repetitions containing all of the elements in $\mathbf{P}^1_\mathcal{R} \setminus \{\top, \bot\}$ and $\mathbf{P}^2_\mathcal{R} \setminus \{\approx\}$, respectively)

et al., 2009]. In contrast to tableau approaches, however, we cannot directly disallow the application of rule (9) (which would require non-monotonic negation). Instead, when we detect that two elements x and y are of the same type (17), we reuse all of the successors of either term that are introduced to satisfy rules of the form (9) to satisfy the same rules over the other element (18). Since we apply Datalog$^\vee$ rules with higher priority (Definition 3), rules (17) and (18) are preferred over rules of type (9). Therefore, we will not introduce a successor for x to satisfy an existential restriction if we already have introduced one such successor for y. This is essential for termination. Removing the rules (13)–(18) would still yield a correct rewriting from $\mathcal{ALCHIQ}$ into Datalog$^{\vee\exists}$, but the resulting rule set would not be terminating.

Another peculiarity is the ternary predicate Copy, which is a weaker form of Eq that we use to handle functionality on unnamed individuals (10). If $\mathsf{Copy}(x,y,z)$ holds, then all unary relations of y and all binary relations between x and y are copied to z (rules (11) and (12)). Asserting $\mathsf{Eq}(y,z)$ would copy additional binary relations of y and other elements, which would not be sound in our case.

Rules in $\mathsf{rew}^{\vee\exists}(\mathcal{R})$ are of bounded size and can be computed in polynomial time. To establish Theorem 5, we further show that the rewriting proposed in Definition 5 is sound (Lemma 6) and complete (Lemma 7), and that $\mathsf{rew}^{\vee\exists}(\mathcal{R})$ is terminating (Lemma 8).

Lemma 6. *Given a fact set $\mathcal{F}$ and a fact φ defined over $\mathbf{P}_\mathcal{R} \setminus \{\top, \bot, \approx\}$, $\langle \mathcal{R}, \mathcal{F}\rangle \not\models \varphi$ implies $\langle \mathsf{rew}^{\vee\exists}(\mathcal{R}), \mathcal{F}\rangle \not\models \varphi$.*

Proof. If $\mathcal{O} = \langle \mathcal{R}, \mathcal{F}\rangle$ does not entail φ, then there is a model $\mathcal{M} \models \mathcal{O}$ such that $\mathcal{M} \not\models \varphi$. To show that $\langle \mathsf{rew}^{\vee\exists}(\mathcal{R}), \mathcal{F}\rangle \not\models \varphi$, we define a model $\mathcal{I} \models \mathsf{rew}^{\vee\exists}(\mathcal{R}) \cup \mathcal{F}$ with $\mathcal{I} \not\models \varphi$.

We define $\mathcal{I}$ via the application of an iterative construction. Initially, we set $\Delta^\mathcal{I} = \{a \mid a \in \mathbf{C}_\mathcal{O}\}$ with $a^\mathcal{I} = a$ (an arbitrary element is used if $\mathbf{C}_\mathcal{O} = \emptyset$). Now, for all $a, b \in \mathbf{C}_\mathcal{O}$: (i) $a^\mathcal{I} \in \mathsf{Named}^\mathcal{I} \cap \mathsf{Top}^\mathcal{I}$; (ii) $a^\mathcal{I} \in A^\mathcal{I} \cap \mathbb{A}^\mathcal{I}$ if $a^\mathcal{M} \in A^\mathcal{M}$ for all $A \in \mathbf{P}^1_\mathcal{R}$; (iii) $\langle a^\mathcal{I}, b^\mathcal{I}\rangle \in R^\mathcal{I} \cap \mathbb{R}^\mathcal{I}$ if $\langle a^\mathcal{M}, b^\mathcal{M}\rangle \in R^\mathcal{M}$ for all $R \in \mathbf{P}^2_\mathcal{R} \setminus \{\approx\}$; and (iv) $\langle a^\mathcal{I}, b^\mathcal{I}\rangle \in \mathsf{Eq}^\mathcal{I}$ if $a^\mathcal{M} = b^\mathcal{M}$.

Furthermore, we extend $\mathcal{I}$ by applying the following 3-step expansion rule once to each element $\gamma \in \Delta^\mathcal{I}$:

- Choose some $\gamma \in \Delta^\mathcal{I}$. Then, $\gamma = a^\mathcal{I}$ for some $a \in \mathbf{C}_\mathcal{O}$ or γ is a domain element of the form $a.\delta_1. \ldots .\delta_k$ with $a \in \mathbf{C}$ and $\delta_1, \ldots, \delta_k \in \Delta^\mathcal{M}$. Let $d(\gamma) = a^\mathcal{M}$ if $\gamma = a^\mathcal{I}$ and $d(\gamma) = \delta_k$ otherwise.
- For every $\rho = A(x) \to \exists y.R(x,y) \wedge B(y) \in \mathcal{R}_\exists$ with $\gamma \in \mathbb{A}^\mathcal{I}$, add $\langle \gamma, \gamma.\epsilon\rangle \in \mathsf{Succ}^\mathcal{I}$ and $\gamma.\epsilon \in \mathsf{Unnamed}^\mathcal{I}$ where $\gamma.\epsilon$ is a fresh null with $\epsilon = \mathit{Sat}^\rho_\mathcal{M}(d(\gamma))$ (i.e., the satisfying successor as defined in the proof of Lemma 3). For all $C \in \mathbf{P}^1_\mathcal{R}$, add $\gamma.\epsilon \in \mathbb{C}^\mathcal{I}$ if $\epsilon \in C^\mathcal{M}$ and $\gamma.\epsilon \in (\mathbb{C}^\neg)^\mathcal{I}$ otherwise. For all $S \in \mathbf{P}^2_\mathcal{R} \setminus \{\approx\}$; add $\langle \gamma, \gamma.\epsilon\rangle \in \mathbb{S}^\mathcal{I}$ if $\langle d(\gamma), \epsilon\rangle \in S^\mathcal{M}$, $\langle \gamma, \gamma.\epsilon\rangle \in (\mathbb{S}^\neg)^\mathcal{I}$ if $\langle d(\gamma), \epsilon\rangle \notin S^\mathcal{M}$, $\langle \gamma.\epsilon, \gamma\rangle \in \mathbb{S}^\mathcal{I}$ if $\langle \epsilon, d(\gamma)\rangle \in S^\mathcal{M}$, and $\langle \gamma.\epsilon, \gamma\rangle \in (\mathbb{S}^\neg)^\mathcal{I}$ if $\langle \epsilon, d(\gamma)\rangle \notin S^\mathcal{M}$.
- Then, minimally extend the function of $\mathcal{I}$ so $\mathcal{I}$ satisfies the Datalog rules (10) and (13–18) from Figure 5.

We can show via induction that, after each extension of $\mathcal{A}$ is computed by applying either of the above expansion rules, the resulting interpretation satisfies all of the Datalog$^\vee$ rules in $\mathsf{rew}^{\vee\exists}(\mathcal{R})$. The induction step can be shown by contradiction: if a Datalog$^\vee$ rule in $\mathsf{rew}^{\vee\exists}(\mathcal{R})$ is not satisfied, then $\mathcal{M}$ is not a model of $\mathcal{O}$. The Datalog$^\exists$ rules in $\mathsf{rew}^{\vee\exists}(\mathcal{R})$ are satisfied by $\mathcal{A}$ because we apply the extension rule fairly. □

Lemma 7. *Given a fact set $\mathcal{F}$ and a fact φ defined over $\mathbf{P}_\mathcal{R} \setminus \{\top, \bot, \approx\}$, $\langle \mathsf{rew}^{\vee\exists}(\mathcal{R}), \mathcal{F}\rangle \not\models \varphi$ implies $\langle \mathcal{R}, \mathcal{F}\rangle \not\models \varphi$.*

Proof. Since $\langle \mathsf{rew}^{\vee\exists}(\mathcal{R}), \mathcal{F}\rangle \not\models \varphi$, there is a model $\mathcal{I} \models \mathsf{rew}^{\vee\exists}(\mathcal{R}) \cup \mathcal{F}$ with $\mathcal{I} \not\models \varphi$. We define a model $\mathcal{M} \models \langle \mathcal{R}, \mathcal{F}\rangle$ with $\mathcal{M} \not\models \varphi$ by an iterative construction.

Initially, let $\mathcal{M} = \mathcal{M}_0$ be as in the proof of Lemma 4. Then, we extend $\mathcal{M}$ by applying extension rules in a fair way. In each step, chose an element $\delta \in \Delta^\mathcal{M}$ for which elements γ, γ', and δ^- can be defined in one of two ways: (1) If $\langle a^\mathcal{I}, \delta\rangle \in \mathsf{Succ}^\mathcal{I}$ for some $a \in \mathbf{C}$, let $\gamma = a^\mathcal{M}$, $\gamma' = a.\delta$, and $\delta^- = a^\mathcal{I}$. (2) If $\langle \delta_k, \delta\rangle \in \mathsf{Succ}^\mathcal{I}$ for some δ_k and some $a.\delta_1.\ldots.\delta_k \in \Delta^\mathcal{M}$, let $\gamma = a.\delta_1.\ldots.\delta_k \in \Delta^\mathcal{M}$, $\gamma' = \gamma.\delta$, and $\delta^- = \delta_k$.

Now add $\gamma' \in \Delta^\mathcal{M}$ as a fresh element, where (i) $\gamma' \in A^\mathcal{M}$ for all $A \in \mathbf{P}^1_\mathcal{R}$ with $\delta \in \mathbb{A}^\mathcal{I}$; and, (ii) for all $R \in \mathbf{P}^2_\mathcal{R}$, let $\langle \gamma, \gamma'\rangle \in R^\mathcal{M}$ if $\langle \delta^-, \delta\rangle \in \mathbb{R}^\mathcal{I}$ and $\langle \gamma', \gamma\rangle \in R^\mathcal{M}$ if $\langle \delta, \delta^-\rangle \in \mathbb{R}^\mathcal{I}$. If γ' is unnecessary in $\mathcal{M}$ (as defined in the proof of Lemma 4), then remove it immediately.

As in Lemma 4, we get a monotonically increasing sequence of interpretations $\mathcal{M}_0, \mathcal{M}_1, \ldots$ with union $\mathcal{M} = \bigcup_{i\geq 1} \mathcal{M}_i$. We can show by induction that each $\mathcal{M}_i$ satisfies the Datalog$^\vee$ rules in $\mathcal{R}$. The Datalog$^\exists$ rules in $\mathcal{R}$ are satisfied by $\mathcal{M}$ because we apply the extension rules fairly. □

Lemma 8. *The rule set $\mathsf{rew}^{\vee\exists}(\mathcal{R})$ is terminating.*

Proof. Consider a set of facts $\mathcal{F}$ and a chase $\mathcal{C}$ of the ontology $\mathcal{O} = \langle \mathsf{rew}^{\vee\exists}(\mathcal{R}), \mathcal{F}\rangle$. We show the lemma by proving that $\mathcal{C}$ is finite. More precisely, we show that $\mathbf{N}_\mathcal{C}$ is finite.

A null $n \in \mathbf{N}_\mathcal{C}$ is *shallow* if $\mathsf{Succ}(a, n) \in \mathcal{C}$ for some constant $a \in \mathbf{C}_\mathcal{O}$; otherwise, n is *deep*. The number of shallow nulls is at most $|\mathcal{R}_\exists| \cdot |\mathbf{C}_\mathcal{O}|$; we proceed to show that the number of deep nulls is finite as well.

For every deep null n, there are some $t_n \in \mathbf{T}_\mathcal{C}$ and $m_n \in \mathbf{N}_\mathcal{C}$ with $\mathsf{Succ}(t_n, m_n), \mathsf{Succ}(m_n, n) \in \mathcal{C}$. Then, the null n is associated with the *type* $\mathit{Typ}(n) = \langle \mathbf{A}, \mathbf{R}, \mathbf{S}, \mathbf{B}\rangle$ where $\mathbf{A} = \{A \mid A(t_n) \in \mathcal{C}, A \in \mathbf{P}^1_\mathcal{R}\}$, $\mathbf{R} = \{R \mid R(t_n, m_n) \in \mathcal{C}, R \in \mathbf{P}^2_\mathcal{R}\}$, $\mathbf{S} = \{R \mid R(m_n, t_n) \in \mathcal{C}, R \in \mathbf{P}^2_\mathcal{R}\}$, and $\mathbf{B} = \{A \mid A(m_n) \in \mathcal{C}, A \in \mathbf{P}^1_\mathcal{R}\}$.

Since Datalog$^\vee$ rules are applied with higher priority, we can show by contradiction that, for every $\rho \in \mathcal{R}_\exists$, there is at most one deep null n associated with any given type $\mathit{Type}(n)$. Since there are $T = 2^{2|\mathbf{P}^1_\mathcal{R}+2|} \cdot 2^{2|\mathbf{P}^2_\mathcal{R}|}$ possible types, there are at most $|\mathcal{R}_\exists| \cdot T$ deep nulls in $\mathcal{C}$. □

5 Horn-$\mathcal{ALCHIQ}$ to Datalog and Datalog$^\exists$

In Sections 3 and 4, we presented rewritings from $\mathcal{ALCHIQ}$ to Datalog$^\vee$ and Datalog$^{\vee\exists}$, respectively. Although we can apply these rewritings to Horn-$\mathcal{ALCHIQ}$, the resulting rule sets will always contain rules with disjunction. We can avoid this by applying dedicated rewritings from Horn-$\mathcal{ALCHIQ}$ to Datalog and Datalog$^\exists$, discussed next.

Ortiz *et al.* [2010] propose a polynomial translation from Horn-$\mathcal{ALCHOIQ}$ – the extension of Horn-$\mathcal{ALCHIQ}$ with nominals – to Datalog^S – an extension of Datalog with set operators – that preserves the outcomes of fact entailment. Furthermore, they discuss how to transform the resulting Datalog^S rule sets into Datalog in polynomial time. Applying this to Horn-$\mathcal{ALCHIQ}$ yields the following result.

Theorem 9. *Horn-$\mathcal{ALCH}(\mathcal{O})\mathcal{IQ}$ is rewritable to Datalog in polynomial time.*

The rewriting shares some commonalities with the type-based method of Section 3, and in particular uses predicates of unbounded (though polynomial) arity for encoding sets.

To get bounded-size rules, we can rewrite to Datalog$^\exists$ using a technique introduced by Carral *et al.* [2019a] for rewriting Horn-$\mathcal{ALC}$ to polynomial, bounded-size, and terminating Datalog$^\exists$. In their approach, an (existing) consequence-based reasoning procedure is expressed in a fixed set of Datalog^S rules, which are then rewritten to terminating Datalog$^\exists$ [Carral *et al.*, 2019a, Theorem 4]. A fundamental difference of this approach to our rewriting in Section 4 is that the whole ontology is rewritten into facts that are combined with a fixed Datalog$^\exists$ rule set. In a similar manner, we adapt an existing consequence-based reasoning procedure for Horn-$\mathcal{ALCHIQ}$ [Eiter *et al.*, 2012] to solve fact entailment (left side of Figure 6) and then implement the adapted calculus using a fixed set of Datalog^S rules (right side of Figure 6). This set can then be translated into a fixed set of terminating Datalog$^\exists$ rules that preserve fact entailment (cf. [Carral *et al.*, 2019a, Theorem 4]). Indeed, this strategy describes how to compute a rewriting from Horn-$\mathcal{ALCHIQ}$ to Datalog$^\exists$, and hence the following holds.

Theorem 10. *Horn-$\mathcal{ALCHIQ}$ is poly-time rewritable to terminating Datalog$^\exists$ rules of bounded size.*

We proceed with the definition of the adapted procedure for solving fact entailment over Horn-$\mathcal{ALCHIQ}$.

Definition 6. *For a Horn-$\mathcal{ALCHIQ}$ ontology $\mathcal{O}$, let $\Xi(\mathcal{O})$ be the set of axioms that results from exhaustively applying the rules in the left hand side of Figure 6 to the axioms in $\mathcal{O}$.*

Note that the above procedure produces DL axioms that are not in the normal form considered in Figure 1. For instance, note the axioms of the form $\mathbb{M} \sqsubseteq \exists\mathbb{S}.\mathbb{N}$ where $\mathbb{M}$ and $\mathbb{N}$ are conjunctions of concept names and $\mathbb{S}$ is a conjunction of roles. The semantics of these axioms are defined as in, e.g., [Eiter *et al.*, 2012].

Theorem 11. *A Horn-$\mathcal{ALCHIQ}$ ontology $\mathcal{O}$ entails a fact φ if and only if $\Xi(\mathcal{O})$ contains φ or a fact of the form $\bot(a)$.*

Now, we describe how to implement the procedure from Definition 6 using a fixed set of Datalog^S rules. For an extended definition of the syntax and semantics of this extension of Datalog, see [Carral *et al.*, 2019a, Section 2].

Definition 7. *For a Horn-$\mathcal{ALCHIQ}$ rule set $\mathcal{R}$, let $\mathsf{rew}^S(\mathcal{R})$ be the (minimal) set of Datalog^S rules such that:*

- *For every axiom in $\mathcal{R}$ of every type defined in Figure 1, the rule set $\mathsf{rew}^S(\mathcal{R})$ contains the corresponding facts:*

  ($\sqcap$) $\mathsf{SC}(\{c_A, c_B\}, c_C)$

$\dfrac{\mathbb{M}(a) \quad \mathbb{M}'(a)}{(\mathbb{M} \sqcap \mathbb{M}')(a) \quad \top(a)}$	$\mathsf{ISA}(a, \mathbb{M}) \wedge \mathsf{ISA}(a, \mathbb{M}') \rightarrow \mathsf{ISA}(a, \mathbb{M} \cup \mathbb{M}') \wedge \mathsf{ISA}(a, \{\top\})$
$\dfrac{\mathbb{S}(a,b) \quad \mathbb{S}'(a,b)}{(\mathbb{S} \sqcap \mathbb{S}')(a,b) \quad \top(a) \quad \top(b)}$	$\mathsf{SPO}(a, \mathbb{S}, b) \wedge \mathsf{SPO}(a, \mathbb{S}', b) \rightarrow \mathsf{SPO}(a, \mathbb{S} \cup \mathbb{S}', b) \wedge \mathsf{ISA}(a, \{\top\}) \wedge \mathsf{ISA}(b, \{\top\})$
$\dfrac{V(a,b)}{\mathit{Inv}(V)(b,a)}$	$\mathsf{SPO}(a, \{V\}, b) \wedge \mathsf{Inv}(V, V_{\mathit{inv}}) \rightarrow \mathsf{SPO}(b, \{V_{\mathit{inv}}\}, a)$
$\dfrac{\mathbb{M}(a) \quad \mathbb{M} \sqsubseteq B}{B(a)}$	$\mathsf{ISA}(a, \mathbb{M}) \wedge \mathsf{SC}(\mathbb{M}, B) \rightarrow \mathsf{ISA}(a, \{B\})$
$\dfrac{\mathbb{M} \sqsubseteq \exists \mathbb{S}.(\mathbb{N} \sqcap \mathbb{N}') \quad \mathbb{N} \sqsubseteq A}{\mathbb{M} \sqsubseteq \exists \mathbb{S}.(\mathbb{N} \sqcap \mathbb{N}' \sqcap A)}$	$\mathsf{Ex}(\mathbb{M}, \mathbb{S}, \mathbb{N} \cup \mathbb{N}') \wedge \mathsf{SC}(\mathbb{N}, A) \rightarrow \mathsf{Ex}(\mathbb{M}, \mathbb{S}, \mathbb{N} \cup \mathbb{N}' \cup \{A\})$
$\dfrac{\mathbb{S}(a,b) \quad \mathbb{S} \sqsubseteq R}{R(a,b)}$	$\mathsf{SPO}(a, \mathbb{S}, b) \wedge \mathsf{SR}(\mathbb{S}, R) \rightarrow \mathsf{SPO}(a, R, b)$
$\dfrac{\mathbb{M} \sqsubseteq \exists(\mathbb{S} \sqcap \mathbb{S}').\mathbb{N} \quad \mathbb{S} \sqsubseteq R}{\mathbb{M} \sqsubseteq \exists(\mathbb{S} \sqcap \mathbb{S}' \sqcap R).\mathbb{N}}$	$\mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \mathbb{S}', \mathbb{N}) \wedge \mathsf{SR}(\mathbb{S}, R) \rightarrow \mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \mathbb{S}' \cup \{R\}, \mathbb{N})$
$\dfrac{\mathbb{M} \sqsubseteq \exists \mathbb{S}.(\mathbb{N} \sqcap \bot)}{\mathbb{M} \sqsubseteq \bot}$	$\mathsf{Ex}(\mathbb{M}, \mathbb{S}, \mathbb{N} \cup \{\bot\}) \rightarrow \mathsf{SC}(\mathbb{M}, \bot)$
$\dfrac{\mathbb{M} \sqsubseteq \exists(\mathbb{S} \sqcap R).\mathbb{N} \quad A \sqsubseteq \forall R.B}{\mathbb{M} \sqcap A \sqsubseteq \exists(\mathbb{S} \sqcap R).(\mathbb{N} \sqcap B)}$	$\mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \{R\}, \mathbb{N}) \wedge \mathsf{Univ}(A, R, B) \rightarrow \mathsf{Ex}(\mathbb{M} \cup \{A\}, \mathbb{S} \cup \{R\}, \mathbb{N} \cup \{B\})$
$\dfrac{A(a) \quad R(a,b) \quad A \sqsubseteq \forall R.B}{B(b)}$	$\mathsf{ISA}(a, \{A\}) \wedge \mathsf{SPO}(a, \{R\}, b) \wedge \mathsf{Univ}(A, R, B) \rightarrow \mathsf{ISA}(b, \{B\})$
$\dfrac{\mathbb{M} \sqsubseteq \exists(\mathbb{S} \sqcap \mathit{Inv}(R)).(\mathbb{N} \sqcap A) \quad A \sqsubseteq \forall R.B}{\mathbb{M} \sqsubseteq B}$	$\mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \{R_{\mathit{inv}}\}, \mathbb{N} \cup \{A\}) \wedge \mathsf{Univ}(A, R, B) \wedge \mathsf{Inv}(R, R_{\mathit{inv}}) \rightarrow \mathsf{SC}(\mathbb{M}, B)$
$\dfrac{A(a) \quad R(a,b) \quad B(b) \quad A \circ R \circ B \sqsubseteq S}{S(a,b)}$	$\mathsf{ISA}(a, \{A\}) \wedge \mathsf{SPO}(a, \{R\}, b) \wedge \mathsf{Circ}(A, R, B, S) \rightarrow \mathsf{SPO}(a, \{S\}, b)$
$\dfrac{\mathbb{M} \sqsubseteq \exists(\mathbb{S} \sqcap R).(\mathbb{N} \sqcap B) \quad A \circ R \circ B \sqsubseteq S}{\mathbb{M} \sqcap A \sqsubseteq \exists(\mathbb{S} \sqcap R \sqcap S).(\mathbb{N} \sqcap B)}$	$\mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \{R\}, \mathbb{N} \cup \{B\}) \wedge \mathsf{Circ}(A, R, B, S) \rightarrow \mathsf{Ex}(\mathbb{M} \cup \{A\}, \mathbb{S} \cup \{R, S\}, \mathbb{N} \cup \{B\})$
$\dfrac{\mathbb{M} \sqsubseteq \exists(\mathbb{S} \sqcap \mathit{Inv}(R)).(\mathbb{N} \sqcap A) \quad A \circ R \circ B \sqsubseteq S}{\mathbb{M} \sqcap B \sqsubseteq \exists(\mathbb{S} \sqcap \mathit{Inv}(R) \sqcap S).(\mathbb{N} \sqcap A)}$	$\mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \{R_{\mathit{inv}}\}, \mathbb{N} \cup \{A\}) \wedge \mathsf{Circ}(A, R, B, S) \wedge \mathsf{Inv}(R, R_{\mathit{inv}}) \rightarrow \mathsf{Ex}(\mathbb{M} \cup \{B\}, \mathbb{S} \cup \{R_{\mathit{inv}}, S\}, \mathbb{N} \cup \{A\})$
$\dfrac{\begin{array}{c} R(a,b) \quad R(a,c) \quad \top \sqsubseteq \leqslant 1\, R.\top \\ C(b) \;/\; V(d,b) \;/\; V(b,d) \end{array}}{C(c) \;/\; V(d,c) \;/\; V(c,d)}$	$\mathsf{SPO}(a, \{R\}, b) \wedge \mathsf{SPO}(a, \{R\}, c) \wedge \mathsf{Funct}(R) \wedge \mathsf{ISA}(b, \{C\}) \;/\; \mathsf{SPO}(d, \{V\}, b) \;/\; \mathsf{SPO}(b, \{V\}, d) \rightarrow \mathsf{ISA}(c, \{A\}) \;/\; \mathsf{SPO}(d, \{V\}, c) \;/\; \mathsf{SPO}(c, \{V\}, d)$
$\dfrac{\begin{array}{c} \mathbb{M}(a) \quad R(a,b) \quad \top \sqsubseteq \leqslant 1\, R.\top \\ \mathbb{M} \sqsubseteq (\mathbb{S} \sqcap R \sqcap V).(\mathbb{N} \sqcap C) \end{array}}{V(a,b) \quad C(b)}$	$\mathsf{ISA}(a, \mathbb{M}) \wedge \mathsf{SPO}(a, \{R\}, b) \wedge \mathsf{Funct}(R) \wedge \mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \{R, V\}, \mathbb{N} \cup \{C\}) \rightarrow \mathsf{SPO}(a, \{V\}, b) \wedge \mathsf{ISA}(b, \{C\})$
$\dfrac{\begin{array}{c} \mathbb{M} \sqsubseteq \exists(\mathbb{S} \sqcap R).\mathbb{N} \quad \top \sqsubseteq \leqslant 1\, R.\top \\ \mathbb{M}' \sqsubseteq \exists(\mathbb{S}' \sqcap R).\mathbb{N}' \end{array}}{\mathbb{M} \sqcap \mathbb{M}' \sqsubseteq \exists(\mathbb{S} \sqcap \mathbb{S}' \sqcap R).(\mathbb{N} \sqcap \mathbb{N}')}$	$\mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \{R\}, \mathbb{N}) \wedge \mathsf{Funct}(R) \wedge \mathsf{Ex}(\mathbb{M}', \mathbb{S}' \cup \{R\}, \mathbb{N}') \rightarrow \mathsf{Ex}(\mathbb{M} \cup \mathbb{M}', \mathbb{S} \cup \mathbb{S}' \cup \{R\}, \mathbb{N} \cup \mathbb{N}')$
$\dfrac{\begin{array}{c} \mathbb{M} \sqsubseteq \exists(\mathbb{S} \sqcap \mathit{Inv}(R)).(\mathbb{N}_1 \sqcap \mathbb{N}_2) \quad \top \sqsubseteq \leqslant 1\, R.\top \\ \mathbb{N}_1 \sqsubseteq \exists(\mathbb{S}' \sqcap R \sqcap V).(\mathbb{N}' \sqcap C) \end{array}}{\mathbb{M} \sqsubseteq \exists(\mathbb{S} \sqcap \mathit{Inv}(R) \sqcap \mathit{Inv}(V)).(\mathbb{N}_1 \sqcap \mathbb{N}_2) \quad \mathbb{M} \sqsubseteq C}$	$\mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \{R_{\mathit{inv}}\}, \mathbb{N}_1 \cup \mathbb{N}_2) \wedge \mathsf{Funct}(R) \wedge \mathsf{Ex}(\mathbb{N}_1, \mathbb{S}' \cup \{R, V\}, \mathbb{N}' \cup \{C\}) \wedge \mathsf{Inv}(R, R_{\mathit{inv}}) \wedge \mathsf{Inv}(V, V_{\mathit{inv}}) \rightarrow \mathsf{Ex}(\mathbb{M}, \mathbb{S} \cup \{R_{\mathit{inv}}, V_{\mathit{inv}}\}, \mathbb{N}_1 \cup \mathbb{N}_2) \wedge \mathsf{SC}(\mathbb{M}, C)$

Figure 6: Horn-$\mathcal{ALCHIQ}$ consequence-based calculus for fact entailment (left) and corresponding Datalog^S rules (right); where *Inv* is a function that maps each role to its inverse; on the left, $\mathbb{M}$, $\mathbb{M}'$, $\mathbb{N}_{(i)}$, and $\mathbb{N}'$ are conjunctions of concept names, A, B, and C are concepts, $\mathbb{S}$ and $\mathbb{S}'$ are conjunctions of roles, R, S, and V are roles; and a, b, and c are individuals; and, on the right, $\mathbb{M}$, $\mathbb{M}'$, $\mathbb{N}_{(i)}$, $\mathbb{N}'$, $\mathbb{S}$, and $\mathbb{S}'$ are set variables, and A, B, C, $R_{(\mathit{inv})}$, S, $V_{(\mathit{inv})}$, a, b, and c are object variables

($\forall$) *Univ*(c_A, c_R, c_B)
($\exists$) *Ex*$(\{c_A\}, \{c_R\}, \{c_B\})$
($\circ$) *Circ*(c_A, c_R, c_B, c_S)
($\leqslant$) *Funct*(c_R)
($\mathbb{R}$) *SR*$(\{c_R, c_S\}, c_V)$ *and* *SR*$(\{c_{R^-}, c_{S^-}\}, c_{V^-})$
($-$) *SR*$(\{c_{R^-}\}, c_S)$ *and* *SR*$(\{c_R\}, c_{S^-})$

In the above, c_A, c_B, c_C, c_R, c_{R^-}, c_S, c_{S^-}, c_V, and c_{V^-} are fresh constants specific to A, B, C, R, S, and V.

- *For every binary predicate R occurring in $\mathcal{R}$, we have that* *Inv*(c_R, c_{R^-}), *Inv*$(c_{R^-}, c_R) \in$ *rew*$^S(\mathcal{R})$.
- *The rule set* *rew*$^S(\mathcal{R})$ *contains the rules in the right hand side of Figure 6.*
- *The rule set* *rew*$^S(\mathcal{R})$ *contains the following rules instantiated for all unary A and binary R predicates occurring in $\mathcal{R}$:*

$$\begin{aligned} A(x) &\to \mathit{ISA}(x, \{c_A\}) \\ R(x,y) &\to \mathit{SPO}(x, \{c_R\}, y) \\ \mathit{ISA}(x, \mathbb{M} \cup \{c_A\}) &\to A(x) \\ \mathit{SPO}(x, \mathbb{S} \cup \{c_R\}, y) &\to R(x,y) \\ \bot(x) \wedge \top(y) &\to A(y) \\ \bot(x) \wedge \top(y) \wedge \top(z) &\to R(y,z) \end{aligned}$$

In the above, x, y, and z are object variables, $\mathbb{M}$ and $\mathbb{S}$ are set variables, and c_A and c_R are constants specific to A and R, respectively.

Theorem 12. *Consider a Horn-$\mathcal{ALCHIQ}$ rule set $\mathcal{R}$. Then,* *rew*$^S(\mathcal{R})$ *is a (Datalog^S) rewriting for $\mathcal{R}$.*

Proof. For a given set of facts $\mathcal{F}$ and a fact φ defined over the signature of $\mathcal{R}$, we can show via induction that φ is entailed by $\langle \mathsf{rew}^S(\mathcal{R}), \mathcal{F}\rangle$ if and only if $\Xi(\langle \mathcal{R}, \mathcal{F}\rangle)$ contains φ or some fact over the special predicate $\bot$. Therefore, the claim follows by Theorem 11. □

As previously discussed, we can transform any Datalog^S rule set into a terminating Datalog$^\exists$ rule set that is polynomial and preserves the outcomes of fact entailment (cf. [Carral *et al.*, 2019a, Theorem 4]). For a Horn-$\mathcal{ALCHIQ}$ rule set $\mathcal{R}$, we can apply this transformation to the Datalog^S rule set $\mathsf{rew}^S(\mathcal{R})$ to compute a terminating Datalog$^\exists$ rewriting for $\mathcal{R}$.

6 Future Work and Conclusions

We show that $\mathcal{ALCHIQ}$ is poly rewritable to Datalog$^\vee$ and to bounded-size Datalog$^{\vee\exists}$ for which the disjunctive chase terminates when prioritising non-generating rules ("Datalog$^\vee$-first"). As for future work, we aim to develop rewriting techniques for more expressive DL languages, such as $\mathcal{ALCHOIQ}$, and decidable rule languages [Baget *et al.*, 2011; König *et al.*, 2015] to terminating Datalog$^{\vee\exists}$.

Termination of our Datalog$^{\vee\exists}$ rewritings relies on the use of the *restricted chase* with a Datalog$^\vee$-first rule application strategy. The set modelling technique used for Horn-$\mathcal{ALCHIQ}$ (Theorem 10) can be adapted to work for arbitrary strategies, but no such modification is known for our rewriting in Section 4. Bounded-size terminating rewritings may still be achieved, even when using the weaker *Skolem chase* [Marnette, 2009], by limiting the depth of terms in the chase using binary counters. However, this will result in a 2ExpTime procedure, while our rewriting achieves NExpTime (cf. proof of Lemma 5). As for all known polynomial rewritings of non-Horn DLs with ExpTime-complete reasoning problems, these are not worst-case optimal. The quest for optimal rule-based approaches in such cases remains open.

Acknowledgements

We thank Sebastian Rudolph for his helpful comments. This work was partly funded by Deutsche Forschungsgemeinschaft (DFG) grant 389792660 as part of TRR 248 and Emmy Noether grant KR 4381/1-1.

References

[Ahmetaj *et al.*, 2016] Shqiponja Ahmetaj, Magdalena Ortiz, and Mantas Simkus. Polynomial Datalog rewritings for expressive Description Logics with closed predicates. In Subbarao Kambhampati, editor, *Proc. 25th Int. Joint Conf. on Artif. Intell. (IJCAI 2016)*, pages 878–885. IJCAI/AAAI Press, 2016.

[Baget *et al.*, 2011] Jean-François Baget, Marie-Laure Mugnier, Sebastian Rudolph, and Michaël Thomazo. Walking the complexity lines for generalized guarded existential rules. In *Proc. 22nd Int. Joint Conf. on Artif. Intell. (IJCAI 2011)*, pages 712–717, 2011.

[Bellomarini *et al.*, 2018] Luigi Bellomarini, Emanuel Sallinger, and Georg Gottlob. The Vadalog system: Datalog-based reasoning for knowledge graphs. *PVLDB*, 11(9):975–987, 2018.

[Bienvenu *et al.*, 2014] Meghyn Bienvenu, Balder ten Cate, Carsten Lutz, and Frank Wolter. Ontology-based data access: A study through disjunctive Datalog, CSP, and MMSNP. *ACM Transactions of Database Systems*, 39(4):33:1–33:44, 2014.

[Carral and Krötzsch, 2020] David Carral and Markus Krötzsch. Rewriting the description logic $\mathcal{ALCHIQ}$ to disjunctive existential rules. In *Proc. 29th Int. Joint Conf. on Artif. Intell. (IJCAI 2020)*, 2020. To appear.

[Carral *et al.*, 2018] David Carral, Irina Dragoste, and Markus Krötzsch. The combined approach to query answering in Horn-$\mathcal{ALCHOIQ}$. In Michael Thielscher, Francesca Toni, and Frank Wolter, editors, *Proc. 16th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2018)*, pages 339–348. AAAI Press, 2018.

[Carral *et al.*, 2019a] David Carral, Irina Dragoste, Markus Krötzsch, and Christian Lewe. Chasing sets: How to use existential rules for expressive reasoning. In Sarit Kraus, editor, *Proc. 28th Int. Joint Conf. on Artif. Intell. (IJCAI 2019)*, pages 1624–1631. ijcai.org, 2019.

[Carral *et al.*, 2019b] David Carral, Larry González, and Patrick Koopmann. From Horn-$\mathcal{SRIQ}$ to Datalog: A data-independent transformation that preserves assertion entailment. In *Proc. 33rd AAAI Conf. on Artificial Intelligence (AAAI 2019)*, pages 2736–2743. AAAI Press, 2019.

[Eiter *et al.*, 2012] Thomas Eiter, Magdalena Ortiz, Mantas Simkus, Trung-Kien Tran, and Guohui Xiao. Query rewriting for Horn-$\mathcal{SHIQ}$ plus rules. In Jörg Hoffmann and Bart Selman, editors, *Proc. 26th AAAI Conf. on Artificial Intelligence (AAAI 2012)*. AAAI Press, 2012.

[Horrocks *et al.*, 2000] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical reasoning for very expressive Description Logics. *Logic J. of the IGPL*, 8(3):239–263, 2000.

[Hustadt *et al.*, 2007] Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Reasoning in Description Logics by a reduction to disjunctive Datalog. *J. Automated Reasoning*, 39(3):351–384, 2007.

[Kazakov *et al.*, 2013] Yevgeny Kazakov, Markus Krötzsch, and František Simančík. The incredible ELK: From polynomial procedures to efficient reasoning with $\mathcal{EL}$ ontologies. *J. Automated Reasoning*, 53:1–61, 2013.

[König *et al.*, 2015] Mélanie König, Michel Leclère, Marie-Laure Mugnier, and Michaël Thomazo. Sound, complete and minimal UCQ-rewriting for existential rules. *Semantic Web*, 6(5):451–475, 2015.

[Krötzsch, 2011] Markus Krötzsch. Efficient rule-based inferencing for OWL EL. In Toby Walsh, editor, *Proc. 22nd Int. Joint Conf. on Artif. Intell. (IJCAI 2011)*, pages 2668–2673. IJCAI/AAAI, 2011.

[Lifschitz, 2019] Vladimir Lifschitz. *Answer Set Programming*. Springer, 2019.

[Marnette, 2009] Bruno Marnette. Generalized schema-mappings: from termination to tractability. In Jan Paredaens and Jianwen Su, editors, *Proc. 28th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2009)*, pages 13–22. ACM, 2009.

[Motik *et al.*, 2009] Boris Motik, Rob Shearer, and Ian Horrocks. Hypertableau reasoning for description logics. *J. Artif. Intell. Res.*, 36:165–228, 2009.

[Motik *et al.*, 2014] Boris Motik, Yavor Nenov, Robert Piro, Ian Horrocks, and Dan Olteanu. Parallel materialisation of Datalog programs in centralised, main-memory RDF systems. In *Proc. 28th AAAI Conf. on Artificial Intelligence (AAAI 2014)*, pages 129–137, 2014.

[Ortiz *et al.*, 2010] Magdalena Ortiz, Sebastian Rudolph, and Mantas Simkus. Worst-case optimal reasoning for the Horn-DL fragments of OWL 1 and 2. In Fangzhen Lin, Ulrike Sattler, and Miroslaw Truszczynski, editors, *Proc. 12th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2010)*. AAAI Press, 2010.

[Rudolph *et al.*, 2012] Sebastian Rudolph, Markus Krötzsch, and Pascal Hitzler. Type-elimination-based reasoning for the Description Logic $\mathcal{SHIQ}b_s$ using decision diagrams and disjunctive Datalog. *Logical Methods in Computer Science*, 8(1), 2012.

[Simancik, 2012] Frantisek Simancik. Elimination of complex RIAs without automata. In Yevgeny Kazakov, Domenico Lembo, and Frank Wolter, editors, *Proc. 25th Int. Workshop on Description Logics (DL 2012)*, volume 846 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[Urbani *et al.*, 2018] Jacopo Urbani, Markus Krötzsch, Ceriel J. H. Jacobs, Irina Dragoste, and David Carral. Efficient model construction for Horn logic with VLog – system description. In *Proc. 9th Int. Joint Conf. on Automated Reasoning (IJCAR 2018), Held as Part of the Federated Logic Conference (FloC 2018)*, pages 680–688, 2018.