Epistemic Logic Programs: Non-Ground and Counting Complexity

Thomas Eiter[1] , **Johannes K. Fichte**[2] , **Markus Hecher**[3] , **Stefan Woltran**[1]
[1]Institute of Logic and Computation, TU Wien
[2]Department of Computer and Information Science (IDA), Linköping University
[3]Computer Science & Artificial Intelligence Laboratory, Massachusetts Institute of Technology
thomas.eiter@tuwien.ac.at, johannes.fichte@liu.se, hecher@mit.edu stefan.woltran@tuwien.ac.at,

Abstract

Answer Set Programming (ASP) is a prominent problem-modeling and solving framework, whose solutions are called answer sets. Epistemic logic programs (ELP) extend ASP to reason about all or some answer sets. Solutions to an ELP can be seen as consequences over multiple collections of answer sets, known as world views. While the complexity of propositional programs is well studied, the non-ground case remains open.

This paper establishes the complexity of non-ground ELPs. We provide a comprehensive picture for well-known program fragments, which turns out to be complete for the class NEXPTIME with access to oracles up to Σ_2^P. In the quantitative setting, we establish complexity results for counting complexity beyond #EXP. To mitigate high complexity, we establish results in case of bounded predicate arity, reaching up to the fourth level of the polynomial hierarchy. Finally, we provide ETH-tight runtime results for the parameter treewidth, which has applications in quantitative reasoning, where we reason on (marginal) probabilities of epistemic literals.

1 Introduction

Answer set programming (ASP) is a widely applied modeling and solving framework for hard combinatorial problems with roots in non-monotonic reasoning and logic programming [Brewka *et al.*, 2011] and solving in propositional satisfiability [Fichte *et al.*, 2023]. In ASP, knowledge is expressed by means of rules forming a *(logic) program*. Solutions to those programs are sets of atoms known as *answer sets*. Epistemic logic programs (ELPs) [Gelfond, 1991; Kahl *et al.*, 2015; Shen and Eiter, 2016; Truszczyński, 2011a] extend ASP by allowing for modal operators **K** and **M**, which intuitively mean "known" or "provably true" and "possible" or "not provably false", respectively. These operators can be included into a program and allow for reasoning over multiple answer sets. Then, solutions to an ELP are known as world views.

Interestingly, the complexity of decision problems, such as whether a ground ELP admits a world view, or whether a literal is true in all respectively some world view, reaches up to the fourth level of the polynomial hierarchy [Shen and Eiter, 2016]. Despite its hardness in the decision case, also counting world views is of vivid research interest today (see e.g., [Besin *et al.*, 2021]), as it provides the connection of quantitative reasoning for ELPs and computing conditional probabilities by considering the proportion of world views compatible with a set of literals. State-of-the-art systems even allow for solving non-ground programs by either replacing variables with domain constants or structural guided grounding and then employing existing ASP solvers [Cabalar *et al.*, 2020; Besin *et al.*, 2022]. Despite the practical implementations and weak known lower bounds [Dantsin *et al.*, 2001; Eiter *et al.*, 2007], the actual complexity for non-ground ELPs – and thus the capabilites of today's non-ground ELP systems – remained entirely open. In particular, it is not known whether epistemic operators in combination with grounding lead to significant complexity amplifications or whether we see only a mild increase (reflected by a jump of one level in the PH – as in the ground case) compared to standard non-ground ASP.

Contributions. In this paper, we study the precise computational *complexity* of *qualitative and quantitative* decision and reasoning problems for *non-ground ELPs*. Our contributions are detailed below. In addition, Table 1 surveys details and illustrates relations to existing results.

1. We provide a comprehensive picture of the non-ground ELP landscape, including common program fragments. We mitigate complexity by showing how complexity results drop if predicate arities are bounded — a typical assumption for solving.
2. We establish detailed complexity results for counting problems, which enables more fine-grained reasoning. To this end, we lift counting complexity notions to the weak-exponential hierarchy.
3. We analyze the impact of structural restrictions in form of bounded treewidth. If the predicate arities are bounded, we obtain precise upper bounds. Surprisingly, the complexity for tight and normal programs match in the non-ground case, which is different to ground programs. We complete the upper bounds by conditional *lower bounds* assuming *ETH*[1] rendering significant runtime improve-

[1]The exponential time hypothesis (ETH) implies 3-CNF satisfiability cannot be solved in time $2^{o(n)}$ [Impagliazzo and Paturi, 2001].

ments for treewidth very unlikely.

Interestingly, our results are based on two sophisticated techniques. First, a classical technique employing second-order logic with dependencies to descriptive complexity for the qualitative setting. Second, a direct approach relying solely on the validity problem for succinct quantified Boolean formulas, which enables results for the quantitative setting as well as when considering bounded treewidth.

Broader Relation to AI. We see use cases and connections of our results in areas beyond the scope of logic programming. In particular, there are complex challenges in, e.g., conformant planning [Bonet, 2010] or in reasoning modes like abduction [Aliseda, 2017; Eiter and Gottlob, 1995b], which reach the third and the fourth level of the polynomial hierarchy. Such situations can be elegantly modeled via modal operators **K** and **M**, even in the non-ground setting. We expect that the interplay between introspection (i.e., **K** and **M** operator capabilities) and non-ground (first-order-like) rules will be of broader interest, as this is essential to formally model rational agents with different belief sets. Here, we provide precise complexity results, consequences of different modeling features, and insights in parameterized complexity. In addition, with the availability of efficient ELP solvers [Bichler *et al.*, 2020; Cabalar *et al.*, 2020], one obtains a first ELP modeling guide.

Related Work. Eiter *et al.* (2007) establish the computational complexity for qualitative problems of non-ground ASP under bounded predicate arities. For ground ELP, Shen and Eiter (2016) show that qualitative problems are higher up in the Polynomial Hierarchy than for ASP, see Ground case in Table 1. In fact, the central decision problem, checking whether an ELP has a world view, is Σ_3^{P}-complete. For treewidth and ground ELP, there are solvers that exploit treewidth [Bichler *et al.*, 2020; Hecher *et al.*, 2020] and also solvers for quantitative reasoning, which relate the the number of accepting literals to number of compatible world views [Besin *et al.*, 2021]. Very recent works address the grounding bottleneck for solving with ELP solvers by grounding that exploits structure [Besin *et al.*, 2023] and complexity of ground ELP when bounded by treewidth Fandinno and Hecher (2023). Our results reach beyond as we consider the *non-ground quantitative* and *qualitative* setting. While the non-ground case might seem somewhat expected, establishing results on the exponential hierarchy requires different techniques, especially for treewidth. Fichte *et al.* (2022b) consider plausibility reasoning in the ground setting for ASP without epistemic operators.

2 Preliminaries

We assume familiarity with basics in Boolean satisfiability (SAT) [Kleine Büning and Lettman, 1999]. By $\exp(\ell, k)$ we refer to k if $\ell \leq 0$ and to $2^{\exp(\ell-1,k)}$ otherwise.

Computational Complexity. We follow standard notions in computational complexity theory [Papadimitriou, 1994; Arora and Barak, 2009] and use the asymptotic notation $\mathcal{O}(\cdot)$ as usual. Let Σ and Σ' be some finite alphabets. We call $I \in \Sigma^*$ an *instance* and n denotes the size of I. A *decision problem* is some subset $L \subseteq \Sigma^*$. Recall that P and NP are the complexity classes of all deterministically and non-deterministically polynomial-time solvable decision problems [Cook, 1971], respectively. We also need the Polynomial Hierarchy (PH) [Stockmeyer and Meyer, 1973; Stockmeyer, 1976; Wrathall, 1976]. In particular, $\Delta_0^{\mathrm{P}} := \Pi_0^{\mathrm{P}} := \Sigma_0^{\mathrm{P}} := \mathrm{P}$ and $\Delta_i^{\mathrm{P}} := P^{\Sigma_{i-1}^{p}}$, $\Sigma_i^{\mathrm{P}} := \mathrm{NP}^{\Sigma_i^{\mathrm{P}}}$, and $\Pi_i^{\mathrm{P}} := \mathrm{coNP}^{\Sigma_i^{\mathrm{P}}}$ for $i > 0$ where C^D is the class C of decision problems augmented by an oracle for some complete problem in class D. The complexity class $\mathrm{D}_k^{\mathrm{P}}$ is defined as $\mathrm{D}_k^{\mathrm{P}} := \{ L_1 \cap L_2 \mid L_1 \in \Sigma_k^{\mathrm{P}}, L_2 \in \Pi_k^{\mathrm{P}} \}$ and $\mathrm{D}^{\mathrm{P}} = \mathrm{D}_1^{\mathrm{P}}$ [Lohrey and Rosowski, 2023]. The complexity class NEXP is the set of decision problems that can be solved by a non-deterministic Turing machine using time $2^{n^{O(1)}}$, i.e., $\mathsf{NEXPTIME} = \bigcup_{k\in\mathbb{N}} \mathsf{NTIME}(2^{n^k})$. The complexity class $\mathrm{NTIME}(f(n))$ is the set of decision problems that can be solved by a non-deterministic Turing machine which runs in time $\mathcal{O}(f(n))$. Note that CO-NEXP is contained in $\mathrm{NEXP}^{\mathrm{NP}}$. The *weak EXP hierarchy (EXPH)* is defined in terms of oracle complexity classes: $\Sigma_0^{\mathrm{EXP}} := \mathrm{EXP}$ and $\Sigma_{i+1}^{\mathrm{EXP}} := \mathrm{NEXP}^{\Sigma_i^{P}}$ [Hemachandra, 1987]. We follow standard notions in counting complexity [Valiant, 1979; Durand *et al.*, 2005; Hemaspaandra and Vollmer, 1995]. A counting problem is a function $f : \Sigma^* \rightarrow \mathbb{N}_0$. Then, #P is the class of all functions $f : \Sigma^* \rightarrow \mathbb{N}_0$ such that there is a polynomial-time non-deterministic Turing machine M, where for every instance $I \in \Sigma^*$, $f(I)$ outputs the number of accepting paths of the Turning machine's computation graph on input I. We will also make use of classes preceded with the sharp-dot operator '$\#\cdot$' defined using witness functions and respective decision problem in a decision complexity class.

Answer Set Programming (ASP). Let $(\mathcal{P}, \mathcal{C})$ be a first-order vocabulary of non-empty finite sets $\mathcal{P}$ of *predicate* and $\mathcal{C}$ of *constant symbols*, and let V be a set of *variable symbols*. *Atoms* a have the form $p(t_1, \ldots, t_n)$, where $p \in \mathcal{P}$, $n \geq 0$ is the arity of p, and each $t_i \in \mathcal{T}$, where $\mathcal{T} = \mathcal{C} \cup \mathcal{V}$ is the set of *terms*. A *logic program (LP)* is a set P of *rules* r of the form

$$a_1 \vee \ldots \vee a_k \leftarrow a_{k+1}, \ldots, a_m, \neg a_{m+1}, \ldots, \neg a_n,$$

where all a_i are distinct atoms and $0 \leq k \leq m \leq n$. We let $H_r := \{a_1, \ldots, a_k\}$, $B_r^+ := \{a_{k+1}, \ldots, a_m\}$, and $B_r^- := \{a_{m+1}, \ldots, a_n\}$, and denote the set of *atoms* occurring in r and P by $\mathrm{at}(r) := H_r \cup B_r^+ \cup B_r^-$, $\mathrm{at}(P) := \bigcup_{r\in P} \mathrm{at}(r)$, and $\mathrm{pnam}(P) := \{ p \mid p(\cdot) \in \mathrm{at}(P) \}$. A program has bounded arity, if every predicate occurring in P has arity at most m for some arbitrary but fixed constant m. By *bounded arity*, refer to the class of programs that are of bounded arity. A rule r is a *fact* if $B_r^+ \cup B_r^- = \emptyset$; a *constraint* if $H_r = \emptyset$; *positive* if $B_r^- = \emptyset$; and *normal* if $|H_r| \leq 1$. A program P is positive and normal, respectively, if each $r \in P$ has the property. The *(positive) dependency graph* $\mathcal{D}_P$ is the digraph with vertices $\bigcup_{r\in P} H_r \cup B_r^+$, where for each rule $r \in P$ two atoms $a \in B_r^+$ and $b \in H_r$ are joined by edge (a, b). Program P is *tight* if $\mathcal{D}_P$ has no directed cycle [Fages, 1994]. We let Full, Normal, and Tight be the classes of all, normal, and tight programs, respectively. Answer sets are defined via ground programs. The *arguments* and *variables* of an atom $a = p(t_1, \ldots, t_n)$ are the sets $\arg(a) := \{t_1, \ldots, t_n\}$ and $\mathrm{vars}(a) := \arg(a) \cap \mathcal{V}$. This extends to sets A of atoms by $\arg(A) := \bigcup_{a\in A} \arg(a)$ resp.

	EHorn	Tight	Normal	Full	Result
Qualitative					
Ground	P	Σ_2^{P}	Σ_2^{P}	Σ_3^{P}	[Truszczyński, 2011b] [Shen and Eiter, 2016]
Non-Ground	EXP	$\mathrm{NEXP}^{\mathrm{NP}}$	$\mathrm{NEXP}^{\mathrm{NP}}$	$\mathrm{NEXP}^{\boldsymbol{\Sigma}_2^{\mathrm{P}}}$	*Theorem 4, Lemma 5*
Non-Ground(b)	coNP	$\boldsymbol{\Sigma}_3^{\mathrm{P}}$	$\boldsymbol{\Sigma}_3^{\mathrm{P}}$	$\boldsymbol{\Sigma}_4^{\mathrm{P}}$	*Theorem 4, Lemma 5*
Quantitative					
Ground	#P	$\# \cdot \mathrm{D}^{\mathrm{P}}$	$\# \cdot \mathrm{D}^{\mathrm{P}}$	$\# \cdot \mathrm{D}_2^{\mathrm{P}}$	*Lemma 15*
Non-Ground	#EXP	$\#\mathrm{EXP}^{\mathrm{NP}}$	$\#\mathrm{EXP}^{\mathrm{NP}}$	$\#\mathrm{EXP}^{\Sigma_2^{\mathrm{P}}}$	*Theorem 14*
Non-Ground(b)	$\# \cdot \mathrm{D}^{\mathrm{P}}$	$\# \cdot \mathrm{D}_2^{\mathrm{P}}$	$\# \cdot \mathrm{D}_2^{\mathrm{P}}$	$\# \cdot \mathrm{D}_3^{\mathrm{P}}$	*Lemma 16*
Parameterized					
Ground [tw]	$\exp(1, o(tw))^{\dagger}$	$\exp(2, \Theta(tw))$	$\exp(2, \Theta(tw \cdot \log(tw)))$	$\exp(3, \Theta(tw))$	[Fandinno and Hecher, 2023]
Non-Gr. [tw](b)	$\exp(1, \mathbf{d}^{\mathbf{o(tw)}})^{\dagger}$	$\exp(2, \mathbf{d}^{\mathbf{\Theta(tw)}})$	$\exp(2, \mathbf{d}^{\mathbf{\Theta(tw)}})$	$\exp(3, \mathbf{d}^{\mathbf{\Theta(tw)}})$	*Theorems 19,20*

Table 1: Complexity results of WV existence (counting/plausibility level) for ELP fragments, where each column states the corresponding fragment and each row gives the respective problem. "(b)" indicates fixed predicate arities. Entries indicate completeness results, runtimes are tight under ETH, omitting polynomial factors. d refers to the domain size and tw is the treewidth of the primal graph. "†": The runtime bounds are for the counting case, as decision is easier due to classical complexity results. Here $\exp(0, n) = n$ and $\exp(k, n) = 2^{\exp(k-1,n)}$, $k \geq 1$.

$\mathrm{vars}(A) \coloneqq \bigcup_{a \in A} \mathrm{vars}(a)$ and likewise to rules and programs. An atom, rule or program φ is *ground* if $\mathrm{vars}(\varphi) = \emptyset$ and *propositional* if $\arg(\varphi) = \emptyset$. The *Herbrand universe* of a program P is the set $\mathcal{U}_P \coloneqq \arg(P) \cap \mathcal{C}$ (if empty, $\mathcal{U}_P \coloneqq \{c\}$ for any $c \in \mathcal{C}$), and its *Herbrand base* $\mathcal{B}_P$ consists of all ground atoms with a predicate from P and constants from $\mathcal{U}_P$.

A set $M \subseteq \mathcal{B}_P$ of atoms *satisfies* (is a *model of*) a ground rule r resp. ground program P if (i) $(H_r \cup B_r^-) \cap M \neq \emptyset$ or (ii) $B_r^+ \setminus M \neq \emptyset$ resp. M satisfies each $r \in P$. Furthermore, M is an *answer set* of P if M is a $\subseteq$-minimal model of $P^M \coloneqq \bigcup_{r \in P} \{ H_r \leftarrow B_r^+ \mid B_r^- \cap M = \emptyset \}$, i.e., the *GL-reduct* of P [Gelfond and Lifschitz, 1991] w.r.t. M; $AS(P)$ denotes the set of all answer sets of P. The answer sets of a general program P are those of its *grounding* $\mathrm{grd}(P) \coloneqq \bigcup_{r \in P} \mathrm{grd}(r)$, where $\mathrm{grd}(r)$ is the set of all rules obtained by replacing each $v \in \mathrm{vars}(r)$ with some element from $\mathcal{U}_P$. We assume *safety*, i.e., each rule $r \in P$ satisfies $\mathrm{vars}(H_r \cup B_r^-) \subseteq \mathrm{vars}(B_r^+)$. We can ensure it by a unary domain predicate dom with facts $dom(c)$, $c \in \mathcal{U}_P$ and adding $dom(x)$ in the body of r for each $x \in \mathrm{vars}(r)$. To select rules in $\mathrm{grd}(P)$ with the same head D, we define $\mathrm{def}(D, P) \coloneqq \{r \in \mathrm{grd}(P) \mid H_r = D\}$ and to select non-ground rules in P that define atoms with predicate p, we let $\mathrm{pdef}(p, P) \coloneqq \{ r \in P \mid p(t_1, \ldots, t_n) \in H_r \}$. Deciding whether a program P has an answer set (called *consistency*) is Σ_2^{P}-complete for ground programs [Eiter and Gottlob, 1995a] and $\mathrm{NEXP}^{\mathrm{NP}}$ for non-ground programs [Eiter *et al.*, 1994].

Epistemic Logic Programs (ELPs). *Epistemic logic programs* extend LPs with epistemic literals in rule bodies. A *literal* is either an atom a (*positive literal*) or its negation $\neg a$ (*negative literal*). A set L of literals is *consistent* if for every $\ell \in L$, $\neg \ell \notin L$ assuming that $\neg\neg\ell = \ell$. For a set A of atoms, we define $\neg A \coloneqq \{ \neg a \mid a \in A \}$ and $\mathrm{lits}(P) \coloneqq \mathrm{at}(P) \cup \neg\, \mathrm{at}(P)$. An *epistemic literal* an expression $\mathbf{not}\, \ell$ where ℓ is a literal. Following common convention, we use $\mathbf{K}\ell$ as shorthand for $\neg\, \mathbf{not}\, \ell$ and $\mathbf{M}\ell$ for $\mathbf{not}\, \neg \ell$. An *epistemic atom* is an atom that is used in an epistemic literal. For a set S consisting of atoms, literals, and/or epistemic literals, we denote by $\mathrm{at}(S)$ and $\mathrm{lits}(S)$, the set of atoms and literals, respectively, that occur in S. These notations naturally extend to rules and programs. Definitions for logic programs such as classes of programs naturally extend to ELP. The *dependency graph* D_P of an ELP P is as for ASP, but for every rule $r \in P$ and $b \in H_r$, we also add an edge (a, b) if r contains a body literal $\mathbf{not}\, \neg a$ or $\neg\, \mathbf{not}\, a$. Properties are similar to ASP. In addition, we define EHorn with no negations (neither $\neg$ nor **not**) and no disjunctions, however, $\mathbf{K}a$ and $\mathbf{M}a$ are allowed. There are different semantics for ELPs , e.g. [Gelfond, 1991; Truszczyński, 2011b; Kahl *et al.*, 2015; Fariñas del Cerro *et al.*, 2015; Shen and Eiter, 2016]; see [Fandinno *et al.*, 2022] for an overview. We consider [Shen and Eiter, 2016], which provides a reduct-based framework and offers highest problem solving capacity.

In what follows, let P be a ground ELP. A *world view interpretation* (*WVI*) for P is a consistent set $I \subseteq \mathrm{lits}(\mathsf{P})$. Intuitively, every $\ell \in I$ is considered "known" and every $a \in \mathrm{at}(\mathsf{P})$ with $\{a, \neg a\} \cap I = \emptyset$ is treated as "possible". The *epistemic reduct* [Shen and Eiter, 2016; Morak, 2019] of program P under WVI I is $\mathsf{P}^I \coloneqq \{r^I \mid r \in \mathsf{P}\}$, where r^I results by replacing in r each epistemic literal $\mathbf{not}\, \ell$ with $\neg \ell$ if $\ell \in I$ and with $\top$ otherwise; double negation cancels. This amounts to FLP-semantics for nested negation; we omit HT-semantics, for which similar complexity results can be obtained. Note that P^I has no epistemic negations. A WVI I over $\mathrm{lits}(\mathsf{P})$ is *compatible* with a set $\mathcal{I}$ of WVIs if (i) $\mathcal{I} \neq \emptyset$ and for each atom a, (ii) $a \in I$ implies $a \in \bigcap_{J \in \mathcal{I}} J$; (iii) $\neg a \in I$ implies $\{J \in \mathcal{I} \mid a \in J\} = \emptyset$; and (iv) $a \in \mathrm{at}(\mathsf{P}) \setminus \mathrm{at}(I)$ implies that $a \in J$ and $a \notin J'$. for some $J, J' \in \mathcal{I}$. I is a *candidate world view (WV)* of P if I is compatible with the set $AS(\mathsf{P}^I)$. *WV existence* is Σ_3^{P}-complete [Truszczyński, 2011b; Shen and Eiter, 2016]. The counting problem #WV asks to output the number of WVs. Semantics of non-ground ELPs is defined by grounding, as for LPs.

Example 1 (cf. Gelfond 1991)**.** *Take the well-known scholarship eligibility problem encoding, which is as follows:*

$$\begin{aligned} P_1 = \{\ & \mathit{lowGPA}(mark);\quad \mathit{highGPA}(mia); \\ & \mathit{lowGPA}(maya) \vee \mathit{highGPA}(maya); \\ & \mathit{inelig}(X) \leftarrow \mathit{lowGPA}(X); \\ & \mathit{elig}(X) \leftarrow \mathit{highGPA}(X); \\ & \bot \leftarrow \mathit{elig}(X), \mathit{inelig}(X); \\ & \mathit{interview}(X) \leftarrow \mathbf{not}\, \mathit{elig}(X), \mathbf{not}\, \mathit{inelig}(X)\ \}. \end{aligned}$$

Then, the set of WVs of the program is

$$\begin{aligned} \{\ \{ & \neg \mathit{interview}(mark), \mathit{lowGPA}(mark), \mathit{inelig}(mark), \\ & \neg \mathit{elig}(mark), \mathit{interview}(maya), \neg \mathit{interview}(mia), \\ & \mathit{highGPA}(mia), \mathit{elig}(mia), \neg \mathit{inelig}(mia)\}\ \}. \end{aligned}$$

First- and Second-Order Logic. We assume familiarity with logic and follow standard definitions [Grädel *et al.*, 2007] (see also the supplemental material). Throughout we assume that σ is a signature, which we omit if it is clear from context. The class $\Sigma^1_k[\sigma]$ consists of all *prenex second-order formulas* $\Phi \in \mathrm{SO}[\sigma]$, i.e., $\Phi = Q_1 R_1 Q_2 R_2 \cdots Q_k R_k.\varphi$ where $Q_i \in \{\forall, \exists\}$ and $Q_i \neq Q_{i+1}$ for $1 \leq i < k$, the R_i are disjoint non-empty sets of SO-variables, and $\varphi \in \mathrm{FO}[\sigma]$; Φ is *existential* if $Q_1 = \exists$. We say that Φ is in *CDNF* if $\mathrm{free}(\varphi) = \emptyset$ and (i) k is even and $\varphi = \exists \vec{x}\psi$ with ψ in DNF, or (ii) k is odd and $\varphi = \forall \vec{x}\psi$ with ψ in CNF.

3 Complexity of Non-ground ELP Reasoning

In this section, we establish results on the classical complexity of reasoning with non-ground ELPs. Our first insight is on qualitative reasoning. Therefore, we need Proposition 2, which states the relationship between second-order logic and the exponential hierarchy for combined and data complexity.

Proposition 2 (Gottlob *et al.* 1999)**.** *Given a sentence $\Phi \in \Sigma^1_k$ and a finite structure $\mathcal{A}$, deciding whether $\mathcal{A} \models \Phi$ is (i) NEXP$^{\Sigma^P_{k-1}}$-complete (combined complexity) and (ii) Σ^P_k-complete if Φ is fixed (data complexity).*

Next, in Lemma 3, we show a connection between the existing result on second-order logic and the exponential hierarchy in the general case and in case predicates have bounded arity.

Lemma 3 ($\star$[2])**.** *Given a sentence $\Phi \in \Sigma^1_k[\sigma]$ in CDNF and a finite structure $\mathcal{A}$, deciding whether $\mathcal{A} \models \Phi$ is (i) NEXP$^{\Sigma^P_{k-1}}$-complete and (ii) Σ^P_{k+1}-complete if every predicate R_i in Φ has arity at most m for some arbitrary but fixed integer $m \geq 1$.*

3.1 Qualitative Reasoning

With the help of the results above, we are ready to establish the following central insight into the complexity of non-ground ELPs. While it turns out that world view existence on a limited fragment is already complete for a class beyond NEXP, luckily, for bounded predicate arity we obtain completeness results for the fourth level of the polynomial hierarchy.

Theorem 4. *Let P be an ELP and (a) $i = 2$ if $P \in$ Full, and (b) $i = 1$ if $P \in$ Normal $\cup$ Tight. Then, deciding whether P admits a world view is NEXP$^{\Sigma^P_i}$-c for non-ground P and Σ^P_{i+2}-c for non-ground of* bounded arity.

[2] We prove ($\star$)-statements in an extended version [Eiter *et al.*, 2024].

Proof (Sketch). Membership: For the non-ground cases, the result follows immediately from the Σ^p_{i+1}-completeness in the ground (propositional) case [Shen and Eiter, 2016], as grounding an ELP P leads to an exponentially larger program $\mathrm{grd}(\mathsf{P})$ and Σ^p_j becomes NEXP$^{\Sigma^P_{j-1}}$ [Gottlob *et al.*, 1999]. For the bounded arity cases. If predicate arities are bounded by a constant, a guess for a WVI I of an epistemic program P has polynomial size. We can emulate the epistemic reduct P^I by replacing in P each epistemic literal $\mathbf{not}\,\ell$ where $\ell = L(\vec{t})$, $L \in \{p, \neg p\}$ by an atom $q_{\mathbf{not}\,L}(\vec{t})$, where $_{\mathbf{not}\,L}$ is a fresh predicate of arity $|\vec{t}|$, and add the following fact or rule, for each tuple $\vec{c}$ of constants (having arity $|\vec{t}|$: (1) $q_{\mathbf{not}\,L}(\vec{c})$, if $L(\vec{c}) \notin I$, and (2) $q_{\mathbf{not}\,L}(\vec{c}) \leftarrow \neg L(\vec{c})$ otherwise (double negation cancels). Then the answer sets of the resulting program $\mathsf{P}_{I,\mathbf{not}}$ correspond to the answer sets of P^I, as (1) and (2), respectively, can be unfolded with rules in the grounding of P that contain $\mathbf{not}\,L(\vec{c})$. In particular, I is compatible with $AS(\mathsf{P}^I)$ iff I is compatible with $AS(\mathsf{P}_{I,\mathbf{not}})$. As $\mathsf{P}_{I,\mathbf{not}}$ has bounded predicate arity, brave and cautious reasoning from $\mathsf{P}_{I,p_{\mathbf{not}}}$ is in Σ^p_3 and Π^p_3, respectively, [Eiter *et al.*, 2007]. Consequently, we can check in polynomial time with an Σ^p_3 oracle whether I fulfills conditions (i)–(ii) of compatibility with $AS(\mathsf{P}^I)$, i.e., whether I is a WVI of P. This shows membership in Σ^p_4, i.e., $\mathsf{P} \in$ Full. If $P \in$ Tight $\cup$ Normal, brave and cautious reasoning from $\mathsf{P}_{I,p_{\mathbf{not}}}$ is in Σ^p_2 and Π^p_2, respectively, [Eiter *et al.*, 2007], as program $\mathsf{P}_{I,p_{\mathbf{not}}}$ is normal/tight, if program P is so. This shows membership in Σ^p_3.

Hardness: We construct from a given sentence $\Phi \in \Sigma^1_k$ and finite structure $\mathcal{A}$ an ELP P, thereby, we reduce deciding whether $\mathcal{A} \models \Phi$ (model checking) to deciding whether P admits a world view (world-view-existence). In our reduction, we lift the existing ELP encoding that solves QBF validity to SO [Shen and Eiter, 2016]. In detail, Case $P \in$ Full: let $\mathcal{A} = (A, \sigma^{\mathcal{A}})$ and $\Phi = \exists R_1 \forall R_2 \exists R_3.\varphi \in \Sigma^1_k$ where $\varphi = \forall \vec{x}\psi$ and $\psi = \bigwedge_{j=1}^{m} \bigvee_{h=1}^{\ell_j} L_{j,h}$ is in CNF, i.e., as in Lemma 3. We take u and v as propositional atoms and for each relation symbol $R \in R_1 \cup R_2 \cup R_2$, we introduce predicates R and $\overline{R}$. Then, we construct programs $P_1, \ldots, P_5$ as follows. Let $e(\vec{x}) = (e(x_1), e(x_2), \ldots, e(x_m))$ and $e(v) \coloneqq v$, if v is an FO-variable and $e(v) \coloneqq c^{\mathcal{A}}$ if v is a constant symbol. Intuitively, P_1 selects with epistemic negation a candidate world view corresponding to a guess for each relation symbol $R_{1,i}$ in R_1 using an auxiliary relation symbol $\overline{R_{1,i}}$ for its complement.

$$\begin{aligned} P_1 = \{ & R_{1,i}(e(\vec{x}_{1,i})) \leftarrow \mathbf{not}\, \overline{R_{1,i}}(e(\vec{x}_{1,i})); \\ & \overline{R_{1,i}}(e(\vec{x}_{1,i})) \leftarrow \mathbf{not}\, R_{1,i}(e(\vec{x}_{1,i})) \mid R_{1,i} \in R_1\}. \end{aligned}$$

Program P_2 generates then answer sets for each possible relation $R_{2,i}$ in R_2.

$$\begin{aligned} P_2 = \{ & \overline{R_{2,i}}(e(\vec{x}_{2,i})) \leftarrow \neg R_{2,i}(e(\vec{x}_{2,i})); \\ & \overline{R_{2,i}}(e(\vec{x}_{2,i})) \leftarrow \neg \overline{R_{2,i}} \mid R_{2,i} \in R_2\} \end{aligned}$$

Programs P_3 guesses for each such valuation of R_2 a valuation of R_3 such that $\forall \vec{x}\psi$ is true.

$$P_3 = \{R_{3,i}(e(\vec{x}_{3,i})) \vee \overline{R_{3,i}}(e(x_{3,i})) \mid R_{3,i} \in R_3\}$$

The program P_4 checks then using the saturation technique that $\forall \vec{x}\psi$ is not violated, i.e., $\exists \vec{x} \neg \psi$ is false; any violation makes u true and saturates the guess. The last rule in P_4 eliminates the candidate world view if $\neg u$ cannot be derived.

$$P_4 = \{u \leftarrow s(L_{j,1}), \ldots, s(L_{j,\ell_j}) \mid 1 \leq j \leq m\} \cup \{R_{3,i}(e(\vec{x}_{3,i})) \leftarrow u;\ \overline{R_{3,i}}(e(\vec{x}_{3,i})) \leftarrow u \mid R_{3,i} \in R_3\} \cup \{v \leftarrow \textbf{not}\, v, \textbf{not}\, \neg u\}.$$

where $s(R(\cdot)) \coloneqq \neg R(\cdot)$ if $R \notin \sigma$, $s(R(\cdot)) \coloneqq R(\cdot)$ otherwise and $s(\neg R(\cdot)) \coloneqq R(\cdot)$.

Program P_5 represents atoms of the input structure $\mathcal{A}$ as facts.

$$P_5 = \{R_i(\vec{c}) \mid \vec{c} \in R_i^{\mathcal{A}}, \vec{c} \in U^{|\vec{c}|}, 1 \leq i \leq k\}.$$

Notably, we treat equality as the other relations. Finally, we build the program $P = \bigcup_{i=1}^{5} P_i$. Then P is constructible in polynomial time and it has a world view iff $\mathcal{A} \models \Phi$.

Case $P \in$ Tight $\cup$ Normal: We encode evaluating an SO sentence $\Phi = \exists P_1 \forall P_2 \exists \vec{x} \psi$ over $\mathcal{A}$. We assume $\psi = \bigvee_{j=1}^{m} \bigwedge_{h=1}^{\ell_j} L_{j,h}$ is a DNF, drop P_3, and replace P_4 with the following rules:

$$P_4' = \{u \leftarrow L_{j,1}, \ldots, L_{j,\ell_j} \mid 1 \leq j \leq m\} \cup \{v \leftarrow \textbf{not}\, v, \textbf{not}\, u\}.$$

For each valuation of P_1 and P_2, we have then a unique answer set that contains u iff $(\mathcal{A}, R_1^{\mathcal{A}}, R_2^{\mathcal{A}}) \models \exists \vec{x} \psi$. Then $P' = P - (P_3 \cup P_4) \cup P_4'$ has a world view iff $\mathcal{A} \models P$. As P_4' is tight, the reduction applies for tight programs as well. ☐

Lemma 5 ($\star$). *Let $P \in$ EHorn. Then, deciding whether P admits a world view is in P if P is ground,* coNP*-complete if P is non-ground and has bounded arity, and EXP-complete if P is non-ground.*

3.2 Counting Complexity Beyond NEXP

Before we can turn our attention to the quantitative setting, we need to define counting classes for classifying counting problems, whose corresponding decision problems are in $\text{NEXP}^{\mathcal{C}}$ for a decision class $\mathcal{C}$. Following, we provide canonical problems, followed by completeness results for ELPs.

Generalizing Counting Classes. For counting solutions of problems in NEXP, the corresponding counting complexity class #EXP [Papadimitriou and Yannakakis, 1986] has been defined. However, classes based on oracle machine models, as in the $\#\cdot$ complexity classes [Hemaspaandra and Vollmer, 1995] have been left out for exponential time. Below, we generalize counting complexity to the realm of exponential time. This allows us to describe in analogy to decision problems the complexity of counting problems beyond NEXP.

Definition 6 (Exp-Oracle Classes). *Let $\mathcal{C}$ be a decision complexity class. Then, $\#\text{EXP}^{\mathcal{C}}$ is the class of counting problems, whose solution is obtained by counting the number of accepting paths of a non-deterministic Turing machine in exponential time with access to a $\mathcal{C}$ oracle.*

Observe that by construction $\#\text{EXP} = \#\text{EXP}^{\text{P}}$. To demonstrate these classes, we define a family of counting problems serving as canonical representatives.

Succinct Quantified Boolean Formulas. To define succinct formula representation, we vastly follow existing ideas [Williams, 2008]. For a circuit C with a set I of n inputs, where C has size $\text{poly}(n)$, we let $T(C)$ be the *truth table* of the Boolean function represented by C. Formally, $T(C)$ is the 2^n-bit string such that $T(C)[i] = C(B_i)$, where B_i is the i-th of all n-bit strings in lexicographic order; intuitively, $T(C)[i]$ is bit i of a string that encodes a problem instance.

For a 3CNF φ, we define such a circuit C_φ ("*clause circuit*") over *sign-bits* s_j for $j \in [1,3]$ and *variable-bits* b_j^k for $k \in [1, v_\varphi]$, where $v_\varphi = \lceil \log(|\operatorname{var}(\varphi)|) \rceil$. More precisely, C_φ has $3(v_\varphi + 1)$ input bits $s_1, \vec{b_1}, s_2, \vec{b_2}, s_3, \vec{b_3}$, where s_j tells whether the j-th literal ℓ_j in a 3CNF clause, whose variable is encoded by the bits $\vec{b_j} = b_j^1, \ldots, b_j^{v_\varphi}$, is positive ($s_j = 1$) or negative ($s_j = 0$). That is, $\ell_1 \vee \ell_2 \vee \ell_3 \in \varphi$ if and only if $C_\varphi(\text{sgn}(\ell_1), \vec{b_1}, \text{sgn}(\ell_2), \vec{b_2}, \text{sgn}(\ell_3), \vec{b_3}) = 1$. For φ in 3DNF, a circuit C_φ ("*term circuit*") is defined analogously.

Example 7. *Let $(a \vee b \vee \neg c) \wedge (\neg b \vee a \vee d) \wedge (\neg b \vee c \vee \neg d)$ be a Boolean formula in 3CNF. Using $2 = \lceil (\log(4)) \rceil$ bits, we can succinctly represent this formula as a circuit.*

For QBFs, we must also succinctly represent quantifiers. While we could merge this into the clause or term circuit, for the sake of readability, we use a second circuit. For a QBF $Q = \exists V_1.\forall V_2.\ldots Q_\ell V_\ell.\varphi$ with alternating quantifier blocks $Q_i \in \{\exists, \forall\}$, we define a *quantifier circuit* C_Q with $\lceil \log(l) \rceil + v_\varphi$ many input bits $\vec{q}, \vec{b}$, where $\vec{q} = q^1, \ldots, q^{\lceil \log(l) \rceil}$ and $\vec{b} = b^1, \ldots, b^{v_\varphi}$. Intuitively, $v \in \operatorname{var}(\varphi)$ is in V_ι iff $C_Q(bin(\iota), bin(v)) = 1$, where $bin(\cdot)$ is the binary representation. Q is *closed* if every $v \in \operatorname{var}(\varphi)$ is in V_ι for some ι; otherwise Q is *open* and its (set of) *free variables* $\operatorname{var}(\varphi) \setminus (\bigcup_{1 \leq \iota \leq \ell} V_\iota)$.

Definition 8 (Succinct QBF). *A* succinct QBF *Q with ℓ alternating quantifier blocks (alternation depth) is given by a* quantifier circuit *C_Q and a* clause circuit *C_φ. Problem $\textsc{SuccQVal}_\ell$ is deciding whether a closed succinct QBF Q evaluates to true; $\#\textsc{SuccQVal}_\ell$ asks to count assignments θ over the free variables of Q such that $Q\theta$ evaluates to true.*

The following complexity result is known.

Proposition 9 (Complexity of $\textsc{SuccQVal}_\ell$ [Gottlob *et al.*, 1999; Stewart, 1991]). *For succinct QBFs Q of alternation depth $\ell \geq 1$, $\textsc{SuccQVal}_\ell$ is $\text{NEXP}^{\Sigma_{\ell-1}^P}$-complete.*

This immediately yields corresponding counting complexity.

Proposition 10 (Complexity of $\#\textsc{SuccQVal}_\ell$). *For succinct QBFs Q of alternation depth $\ell \geq 0$, counting the number of assignments over its free variables under which Q evaluates to true is $\#\text{EXP}^{\Sigma_\ell^P}$-complete.*

We will utilize this result by defining a parsimonious reduction to our counting problems of interest. A *parsimonious reduction* is a polynomial-time reduction from one problem to another that preserves the number of solutions, i.e., it induces a bijection between respective sets of solutions of two problems.

3.3 Quantitative Reasoning

Next, we discuss how quantitative aspects enable more fine-grained reasoning. Indeed, deciding whether a world view exists concerns only a single world view. Instead, if we aim for stable results towards consensus among different world views, one would prefer computing *levels of plausibility* for certain observations (assumptions). This is achieved by quantitative reasoning, where we quantify the number of world views satisfying a given query Q. Thereby we compute the level of plausibility for Q. We need the following notation.

Definition 11. *An* (epistemic) query *is a set of expressions of the form* $\mathbf{M}\ell$ *or* $\mathbf{K}\ell$*, where* ℓ *is a literal.*

Intuitively, we can then quantify the plausibility of queries. To this end, we define the union of an ELP P and a query Q, which is a set of expressions as defined above, by $\mathsf{P} \sqcup Q \coloneqq \mathsf{P} \cup \{v \leftarrow \textbf{not}\, v, \neg q \mid q \in Q\}$ for a fresh atom v.

Definition 12 (Plausibility Level)**.** *Let* Q *be an epistemic query and* P *be an ELP. The* plausibility level *of* Q *is defined as* $L(\mathsf{P}, Q) \coloneqq \#WV(\mathsf{P} \sqcup Q)$.

We define probabilities via two counting operations and therefore study the complexity of computing plausibility levels.

Definition 13 (Probability)**.** *Let* Q *be an epistemic query and* P *be an ELP. The* probability *of* Q *is defined as* $\frac{L(\mathsf{P},Q)}{\max(1,L(\mathsf{P},\emptyset))}$.

Observe that the empty query has probability 1.0 and inconsistent queries or ELPs both have probability 0.0 (implausible).

Complexity of Computing Plausibility Levels. For establishing the complexity of counting, we reduce from $\#\textsc{SuccQVal}_\ell$ and use Proposition 10. Indeed, computing plausibility levels is already hard for empty queries.

Theorem 14 ($\star$)**.** *Let* P *be an ELP and (a)* $i = 2$ *if* $\mathsf{P} \in$ *Full, (b)* $i = 1$ *if* $\mathsf{P} \in$ *Normal* $\cup$ *Tight, and (c)* $i = 0$ *if* $\mathsf{P} \in$ *EHorn. Then, computing plausibility level* $L(\mathsf{P}, \emptyset)$ *is* $\#\mathrm{EXP}^{\Sigma_i^P}$*-complete.*

Proof. Membership: This follows immediately from the ground case (propositional) where we have Σ_{i+1}^p-completeness [Shen and Eiter, 2016]. With the same argument as in the proof of Theorem 14, meaning, grounding an ELP P leads to an exponentially larger program $\mathrm{grd}(\mathsf{P})$, and Σ_j^p becomes $\mathrm{NEXP}^{\Sigma_{j-1}^{\mathrm{P}}}$ cf. [Gottlob *et al.*, 1999], we conclude the result.

Hardness for $\mathsf{P} \in$ *EHorn:*
We reduce from a *restricted fragment* of $\#\textsc{SuccQVal}_0$, taking a positive Boolean formula φ defined by a clause circuit C over $3 \cdot (1 + n)$ many input gates, and constructing an ELP P.

First, the evaluation of C is inductively constructed, starting from the input gates of C to the output gate of C. Thereby, for every gate g we construct a rule defining a predicate of arity $3 \cdot (1 + n)$, depending on the result of the predicates for the input gates of g. By $\vec{v}_i$ we refer to a sequence of n many variables $v_i^1, \ldots, v_i^n$. Also, we define the facts $b(0)$ and $b(1)$. For an input gate g_j of C with $1 \leq j \leq 3\cdot(1+n)$, we construct the fact $g_j(v^1, \ldots, v^{3\cdot(1+n)})$. if and only if $v_j = 1$. Without loss of generality, we assume that negation only appears at the input gates (negation normal form).

For a conjunction gate $g_\wedge$ with inputs $g_1, \ldots, g_o$, we define
$$g_\wedge(s_1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3) \leftarrow g_1(s_1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3), \ldots, g_o(s_1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3);$$
for disjunction gate $g_\vee$ with inputs $g_1, \ldots, g_o$, we define for every $1 \leq k \leq o$:
$$g_\vee(s_1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3) \leftarrow g_k(s_1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3).$$

We refer to the predicate of the final output gate of the construction by g_C. Additionally, we construct the following rules below. First, we guess an assignment over the variables, where we decide whether a variable will be set to false:
$$\dot{V}(v^1, \ldots, v^n) \leftarrow \mathbf{M}\dot{V}(v^1, \ldots, v^n), b(v^1), \ldots, b(v^n).$$
Then, we check whether there is an unsatisfied clause.
$$\leftarrow \dot{V}(\vec{v}_1), g_C(1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3), \dot{V}(\vec{v}_2), g_C(s_1, \vec{v}_1, 1, \vec{v}_2, s_3, \vec{v}_3), \dot{V}(\vec{v}_3), g_C(s_1, \vec{v}_1, s_2, \vec{v}_2, 1, \vec{v}_3).$$

It is easy to see that there is a bijection between satisfying assignments of φ and world views of P.

Hardness for $\mathsf{P} \in$ *Tight* $\cup$ *Normal*:
We reduce from $\#\textsc{SuccQVal}_1$, taking a QBF $Q = \forall U.\varphi$ over free variables V, with 3DNF φ given by a term circuit C over $3 \cdot (1 + n)$ many input gates and a quantifier circuit D over n input gates. From this, we construct ELP P.

First, C is inductively constructed as above. We refer to the predicate of the output gate of the construction by g_C. Then, similar to above, we define for every gate of the circuit D a predicate of arity $n + 1$ from the input gates to the output gate of D. For a negation gate $g_\neg$ with input g, we define
$$g_\neg(\vec{v}_1) \leftarrow \neg g(\iota, \vec{v}_1), b(\iota), b(v_1^1), \ldots, b(v_1^n);$$
for a conjunction gate $g_\wedge$ with inputs $g_1, \ldots, g_o$, we define
$$g_\wedge(\iota, \vec{v}_1) \leftarrow g_1(\iota, \vec{v}_1), \ldots, g_o(\iota, \vec{v}_1);$$
for disjunction gate $g_\vee$ with input gates $g_1, \ldots, g_o$, we define for every $1 \leq k \leq o$: $\quad g_\vee(\iota, \vec{v}_1) \leftarrow g_k(\iota, \vec{v}_1)$.

The predicate of the final output gate of D is given by g_D.

Additionally, we construct the following rules below, thereby following $\neg\exists U.\overline{\varphi}$ over the inverse formula of φ. First, we guess an assignment over the variables:
$$A(v^1, \ldots, v^n) \leftarrow \textbf{not}\, \dot{A}(v^1, \ldots, v^n), g_D(1, v^1, \ldots, v^n).$$
$$\dot{A}(v^1, \ldots, v^n) \leftarrow \textbf{not}\, A(v^1, \ldots, v^n), g_D(1, v^1, \ldots, v^n).$$
$$A(v^1, \ldots, v^n) \leftarrow \neg\dot{A}(v^1, \ldots, v^n), g_D(2, v^1, \ldots, v^n).$$
$$\dot{A}(v^1, \ldots, v^n) \leftarrow \neg A(v^1, \ldots, v^n), g_D(2, v^1, \ldots, v^n).$$
Then, we check whether all terms are dissatisfied.
$$\mathrm{usat}(1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3) \leftarrow \dot{A}(\vec{v}_1), g_C(1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3)$$
$$\mathrm{usat}(0, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3) \leftarrow A(\vec{v}_1), g_C(0, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3)$$
$$\mathrm{usat}(s_1, \vec{v}_1, 1, \vec{v}_2, s_3, \vec{v}_3) \leftarrow \dot{A}(\vec{v}_2), g_C(s_1, \vec{v}_1, 1, \vec{v}_2, s_3, \vec{v}_3)$$
$$\mathrm{usat}(s_1, \vec{v}_1, 0, \vec{v}_2, s_3, \vec{v}_3) \leftarrow A(\vec{v}_2), g_C(s_1, \vec{v}_1, 0, \vec{v}_2, s_3, \vec{v}_3)$$
$$\mathrm{usat}(s_1, \vec{v}_1, s_2, \vec{v}_2, 1, \vec{v}_3) \leftarrow \dot{A}(\vec{v}_3), g_C(s_1, \vec{v}_1, s_2, \vec{v}_2, 1, \vec{v}_3)$$
$$\mathrm{usat}(s_1, \vec{v}_1, s_2, \vec{v}_2, 0, \vec{v}_3) \leftarrow A(\vec{v}_3), g_C(s_1, \vec{v}_1, s_2, \vec{v}_2, 0, \vec{v}_3).$$
We prohibit WVs with an answer set satisfying a term.
$$\mathrm{sat} \leftarrow g_C(s_1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3), \neg\mathrm{usat}(s_1, \vec{v}_1, s_2, \vec{v}_2, s_3, \vec{v}_3)$$
$$v \leftarrow \textbf{not}\, v, \textbf{not}\, \neg\mathrm{sat}.$$
It is easy to see that there is a bijection between satisfying assignments over V of Q and world views of P. Hardness for normal programs follows immediately from the reduction above, since the resulting programs are already normal.

Hardness for $\mathsf{P} \in$ Full: We provide details in an extended version [Eiter *et al.*, 2024]. ☐

Similarly, we conclude the following statement.

Lemma 15 ($\star$)**.** *Let* P *be a ground ELP and* $i = 2$ *if* $\mathsf{P} \in$ *Full, and* $i = 1$ *if* $\mathsf{P} \in$ *Normal* $\cup$ *Tight, and* $i = 0$ *if* $\mathsf{P} \in$ *EHorn. Then, computing plausibility level* $L(P, \emptyset)$ *is* $\# \cdot \mathrm{D}_i^{\mathrm{P}}$*-complete.*

If the arity is a fixed constant, we obtain the following.

Lemma 16 ($\star$). *Let P be a non-ground ELP of bounded arity and $i = 2$ if $P \in$ Full, and $i = 1$ if $P \in$ Normal $\cup$ Tight, and $i = 0$ if $P \in$ EHorn. Then, computing plausibility level $L(P, \emptyset)$ is $\# \cdot \mathrm{D}^{\mathrm{P}}_{i+1}$-complete.*

4 Non-Ground ELPs of Bounded Treewidth

Before we discuss consequences of evaluating non-ground ELPs for treewidth, we recall tree decompositions (TDs) for which we need the following definition.

Definition 17 (TD [Robertson and Seymour, 1985]). *Let $G = (V, E)$ be a graph. A pair $\mathcal{T} = (T, \chi)$, where T is a rooted tree with root $\mathrm{r}(T)$ and χ is a labeling function that maps every node t of T to a subset $\chi(t) \subseteq V$ called* bag, *is a* tree decomposition (TD) *of G if (i) for each $v \in V$ some t in T exists s.t. $v \in \chi(t)$; (ii) for each $\{v, w\} \in E$ some t in T exists s.t. $\{v, w\} \subseteq \chi(t)$; and (iii) for each r, s, t of T s.t. s lies on the unique path from r to t, $\chi(r) \cap \chi(t) \subseteq \chi(s)$.*

The *width* of $\mathcal{T}$ is the largest bag size minus one and the *treewidth* of G is the smallest width among all TDs of G. To simplify case distinctions in the algorithms, *we use nice TDs in a proof* (see extended version), which can be computed in linear time without increasing the width [Kloks, 1994]. To capture atom (predicate) dependencies of programs, we use the following *primal graph* $G_P = (V, E)$ of a program P defined as follows. For ground P, we let $V := \mathrm{at}(P)$ and $\{a, b\} \in E$ if atoms $a \neq b$ jointly occur in a rule of P, while for non-ground P, we let $V := \mathrm{pnam}(P)$ and $\{p_1, p_2\} \in E$ if predicates $p_1 \neq p_2$ jointly occur in a rule of P. Tree decompositions allow us to establish tight complexity bounds for WV existence under ETH. To this end, we resort to quantified CSP (QCSP), which, intuitively, is analogous to quantified Boolean formulas, but over arbitrary finite domains instead of domain $\{0, 1\}$. To this end, we define primal graph P_Q for a QCSP Q similarly to programs, but on the formula's matrix. Further, $\exp(0, n) = n$ and $\exp(k, n) = 2^{\exp(k-1,n)}$, $k \geq 1$, denotes the k-fold exponential function of n. The following bounds are known.

Proposition 18 (Fichte *et al.* 2020). *Given any QCSP Q with constraints C over finite domain D and alternation depth $\ell \geq 1$, where each constraint has at most $s \geq 3$ variables. Then, under ETH the validity of Q cannot be decided in time $\exp(\ell-1, |D|^{o(k)}) \cdot \mathrm{poly}(|C|)$, where k is the treewidth of P_C.*

With his result at hand, we obtain the following.

Theorem 19 ($\star$). *Let P be an arbitrary ELP of bounded arity a over domain size $d = |\mathrm{dom}(P)|$, where the treewidth of G_P is k. Furthermore, let (a) $i = 2$ if $P \in$ Full, (b) $i = 1$ if $P \in$ Normal $\cup$ Tight, and (c) $i = 0$ if $P \in$ EHorn. Then, under ETH, WV existence of $\mathrm{grd}(P)$ cannot be decided in time $\exp(i + 1, d^{o(k)}) \cdot \mathrm{poly}(|\mathrm{at}(P)|)$.*

Indeed, one can obtain a runtime adhering to this lower bound.

Theorem 20 ($\star$). *Let P be an arbitrary ELP of bounded arity a over domain size $d = |\mathrm{dom}(P)|$, where the treewidth of G_P is k. Furthermore, let (a) $i = 2$ if $P \in$ Full and (b) $i = 1$ if $P \in$ Normal $\cup$ Tight. Then, deciding world view existence as well as computing plausibility level $L(P, \emptyset)$ of $\mathrm{grd}(P)$ can be done in time $\exp(i + 1, d^{\mathcal{O}(k)}) \cdot \mathrm{poly}(|\mathrm{at}(P)|)$.*

5 Conclusion and Outlook

We consider non-ground ELP, a popular concept to enable reasoning about answer sets. We settle the complexity landscape of qualitative and quantitative reasoning tasks for non-ground ELPs, including common program fragments. In particular, we establish that deciding whether a program admits a world view ranges between NEXP and $\mathrm{NEXP}^{\Sigma^{\mathrm{P}}_3}$. We mitigate resulting high complexity by bounding predicate arities. Then, the complexity drops, ranging from Σ^{P}_2 to Σ^{P}_4. In the quantitative setting, we consider levels of plausibility by quantifying the number of world views that satisfy a given query Q. We show completeness results for all common settings and classes of programs, namely, ground programs, non-ground programs, and non-ground programs of bounded arity. We complete these results by incorporating treewidth and establish results ranging up to four-fold exponential runtime in the treewidth, including ETH-tight lower bounds. Due to the techniques, our proofs also work for other common ELP-semantics.

Our results contribute to several avenues for future research. First, we have an indication that well-known problems from the AI domains with high complexity are amenable to ELPs. In particular, we now have an understanding that epistemic operators and fixed predicate arities provide a suitable target formalism for problems on the second, third, or fourth level of the PH, as certain variants of the diagnosis problem [Eiter and Gottlob, 1995b; Eiter *et al.*, 1997], counterfactual reasoning [Eiter and Gottlob, 1996], or default logic [Fichte *et al.*, 2022c]. Modeling such problems using epistemic operators might yield elegant and instructive ASP encodings.Second, they indicate alternative ways for solver design: so far, standard non-ground ELP systems ground the ELP first and then solve the resulting ground ELP. Our results justify that epistemic operators can be reduced on the non-ground level without the exponential blowup. Recall that non-ground, normal ELPs and propositional, disjunctive LPs are of similar complexity (see Table 1). This makes alternative grounding techniques such as lazy grounding [Weinzierl *et al.*, 2020] or body-decoupled grounding [Besin *et al.*, 2022] immediately accessible for ELPs. Also, our results from Section 4 build a theoretical foundation for structure-aware ELP grounders. This could also be interesting for structure-guided reductions to ELP [Hecher, 2022]. Finally, extending the complexity landscape of non-ground ELPs is on our agenda. Finding natural NP-fragments would be interesting, since the complexity beyond EHorn almost immediately jumps two levels in PH for the Shen-Eiter semantics. A comprehensive complexity picture in ELP similar to ASP could be of interest in this setting [Truszczyński, 2011b; Fichte *et al.*, 2015]. We have left aside the case of *maximal* world views so far, although we expect that the complexity increases by one level on the PH for reasoning problems. It might also be interesting to consider complementary aspects in ELPs where modal operators require some literals to be present in answer sets [Fichte *et al.*, 2022a] or where we compute quantitative aspects approximately [Kabir *et al.*, 2022]. Restrictions on epistemic atoms that might be of interest or other structural restrictions on programs, for example, fractional hyper-treewidth [Grohe and Marx, 2014], is subject of future research.

Acknowledgements

Authors are ordered alphabetically. The work has been carried out while Hecher visited the Simons Institute at UC Berkeley. Research is supported by the Austrian Academy of Sciences (ÖAW), DOC Fellowship; the Austrian Science Fund (FWF), grants P30168 and J4656; ELLIIT funded by the Swedish government; Humane AI Net (ICT-48-2020-RIA / 952026); the Society for Research Funding in Lower Austria (GFF) grant ExzF-0004; and Vienna Science and Technology Fund (WWTF) grants ICT19-065 and ICT22-023.

References

[Aliseda, 2017] Atocha Aliseda. *The Logic of Abduction: An Introduction*, pages 219–230. Springer, Cham, 2017.

[Arora and Barak, 2009] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[Besin *et al.*, 2021] Viktor Besin, Markus Hecher, and Stefan Woltran. Utilizing Treewidth for Quantitative Reasoning on Epistemic Logic Programs. *TPLP*, 21(5):575–592, 2021.

[Besin *et al.*, 2022] Viktor Besin, Markus Hecher, and Stefan Woltran. Body-Decoupled Grounding via Solving: A Novel Approach on the ASP Bottleneck. In *IJCAI'22*, pages 2546–2552. ijcai.org, 2022.

[Besin *et al.*, 2023] Viktor Besin, Markus Hecher, and Stefan Woltran. On the structural complexity of grounding - tackling the ASP grounding bottleneck via epistemic programs and treewidth. In *ECAI'23*, pages 247–254. IOS Press, 2023.

[Bichler *et al.*, 2020] Manuel Bichler, Michael Morak, and Stefan Woltran. selp: A single-shot epistemic logic program solver. *TPLP*, 20(4):435–455, 2020.

[Bonet, 2010] Blai Bonet. Conformant plans and beyond: Principles and complexity. *Artificial Intelligence*, 174(3):245–269, 2010.

[Brewka *et al.*, 2011] Gerhard Brewka, Thomas Eiter, and Mirosław Truszczyński. Answer set programming at a glance. *Comm. of the ACM*, 54(12):92–103, 2011.

[Cabalar *et al.*, 2020] Pedro Cabalar, Jorge Fandinno, Javier Garea, Javier Romero, and Torsten Schaub. eclingo : A Solver for Epistemic Logic Programs. *TPLP*, 20(6):834–847, 2020.

[Cook, 1971] Stephen A. Cook. The Complexity of Theorem-Proving Procedures. In *STOC'71*, pages 151–158. ACM, 1971.

[Dantsin *et al.*, 2001] Evgeny Dantsin, Thomas Eiter, Georg Gottlob, and Andrei Voronkov. Complexity and expressive power of logic programming. *ACM Comput. Surv.*, 33(3):374–425, 2001.

[Durand *et al.*, 2005] Arnaud Durand, Miki Hermann, and Phokion G. Kolaitis. Subtractive reductions and complete problems for counting complexity classes. *Theoretical Computer Science*, 340(3):496–513, 2005.

[Eiter and Gottlob, 1995a] Thomas Eiter and Georg Gottlob. On the computational cost of disjunctive logic programming: Propositional case. *Annals of Mathematics and Artificial Intelligence*, 15(3–4):289–323, 1995.

[Eiter and Gottlob, 1995b] Thomas Eiter and Georg Gottlob. The Complexity of Logic-Based Abduction. *Journal of the ACM*, 42(1):3–42, 1995.

[Eiter and Gottlob, 1996] Thomas Eiter and Georg Gottlob. The complexity of nested counterfactuals and iterated knowledge base revisions. *J. Comput. Syst. Sci.*, 53(3):497–512, 1996.

[Eiter *et al.*, 1994] Thomas Eiter, Georg Gottlob, and Heikki Mannila. Adding disjunction to datalog (extended abstract). In *PODS '94*, pages 267–278, New York, NY, USA, 1994. ACM.

[Eiter *et al.*, 1997] Thomas Eiter, Georg Gottlob, and Nicola Leone. Abduction from logic programs: Semantics and complexity. *Theor. Comput. Sci.*, 189(1-2):129–177, 1997.

[Eiter *et al.*, 2007] Thomas Eiter, Wolfgang Faber, Michael Fink, and Stefan Woltran. Complexity results for answer set programming with bounded predicate arities and implications. *Annals of Mathematics and Artificial Intelligence*, 51(2-4):123–165, 2007.

[Eiter *et al.*, 2024] Thomas Eiter, Johannes K. Fichte, Markus Hecher, and Stefan Woltran. Epistemic logic programs: Non-ground and counting complexity. *CoRR*, 2024. Extended Version of a paper that has been accepted for publication at IJCAI'24.

[Fages, 1994] François Fages. Consistency of Clark's completion and existence of stable models. *Logical Methods in Computer Science*, 1(1):51–60, 1994.

[Fandinno and Hecher, 2023] Jorge Fandinno and Markus Hecher. Treewidth-aware complexity for evaluating epistemic logic programs. In *IJCAI'23*, pages 3203–3211. ijcai.org, 2023.

[Fandinno *et al.*, 2022] Jorge Fandinno, Wolfgang Faber, and Michael Gelfond. Thirty years of epistemic specifications. *TPLP*, pages 1043–1083, 2022.

[Fariñas del Cerro *et al.*, 2015] Luis Fariñas del Cerro, Andreas Herzig, and Ezgi Iraz Su. Epistemic equilibrium logic. In *IJCAI*, pages 2964–2970, 2015.

[Fichte *et al.*, 2015] Johannes K. Fichte, Mirosław Truszczyński, and Stefan Woltran. Dual-normal logic programs – the forgotten class. *TPLP*, 15(4–5):495–510, 2015.

[Fichte *et al.*, 2020] Johannes Klaus Fichte, Markus Hecher, and Maximilian F. I. Kieler. Treewidth-Aware Quantifier Elimination and Expansion for QCSP. In *CP'20*, pages 248–266. Springer, 2020.

[Fichte *et al.*, 2022a] Johannes K Fichte, Sarah Alice Gaggl, and Dominik Rusovac. Rushing and strolling among answer sets – navigation made easy. In *AAAI'2022*, 2022.

[Fichte *et al.*, 2022b] Johannes K. Fichte, Markus Hecher, and Mohamed A. Nadeem. Plausibility reasoning via projected answer set counting - a hybrid approach. In *IJCAI'22*, pages 2620–2626. IJCAI Organization, July 2022.

[Fichte *et al.*, 2022c] Johannes K. Fichte, Markus Hecher, and Irina Schindler. Default logic and bounded treewidth. *Information and Computation*, 2022.

[Fichte *et al.*, 2023] Johannes K. Fichte, Daniel Le Berre, Markus Hecher, and Stefan Szeider. The silent (r)evolution of sat. *Commun. ACM*, 66(6):64–72, May 2023.

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9(3/4):365–386, 1991.

[Gelfond, 1991] Michael Gelfond. Strong Introspection. In *AAAI'91*, pages 386–391. AAAI Press / The MIT Press, 1991.

[Gottlob *et al.*, 1999] Georg Gottlob, Nicola Leone, and Helmut Veith. Succinctness as a source of complexity in logical formalisms. *Annals of Pure and Applied Logic*, pages 231–260, 1999.

[Grädel *et al.*, 2007] Erich Grädel, Phokion G Kolaitis, Leonid Libkin, Maarten Marx, Joel Spencer, Moshe Y Vardi, Yde Venema, Scott Weinstein, et al. *Finite Model Theory and its applications*. Springer, 2007.

[Grohe and Marx, 2014] Martin Grohe and Dániel Marx. Constraint solving via fractional edge covers. *ACM Transactions on Algorithms (TALG)*, 11(1):1–20, 2014.

[Hecher *et al.*, 2020] Markus Hecher, Michael Morak, and Stefan Woltran. Structural Decompositions of Epistemic Logic Programs. In *AAAI'20*, pages 2830–2837. AAAI Press, 2020.

[Hecher, 2022] Markus Hecher. Treewidth-aware reductions of normal ASP to SAT – is normal ASP harder than SAT after all? *Artificial Intelligence*, 304:103651, 2022.

[Hemachandra, 1987] Lane A. Hemachandra. The strong exponential hierarchy collapses. In *STOC'87*, pages 110–122, New York, New York, USA, 1987. ACM.

[Hemaspaandra and Vollmer, 1995] Lane A. Hemaspaandra and Heribert Vollmer. The Satanic Notations: Counting Classes Beyond #P and Other Definitional Adventures. *SIGACT News*, 26(1):2–13, March 1995.

[Impagliazzo and Paturi, 2001] Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001.

[Kabir *et al.*, 2022] Mohimenul Kabir, Flavio O Everardo, Ankit K Shukla, Markus Hecher, Johannes Klaus Fichte, and Kuldeep S Meel. ApproxASP – a scalable approximate answer set counter. In *AAAI'22*, pages 5755–5764. AAAI Press, 2022.

[Kahl *et al.*, 2015] Patrick Thor Kahl, Richard Watson, Evgenii Balai, Michael Gelfond, and Yuanlin Zhang. The Language of Epistemic Specifications (Refined) Including a Prototype Solver. *J. Logic Comput.*, 25, 2015.

[Kleine Büning and Lettman, 1999] Hans Kleine Büning and Theodor Lettman. *Propositional Logic: Deduction and Algorithms*, volume 48 of *Cambridge tracts in theoretical computer science*. Cambridge University Press, 1999.

[Kloks, 1994] Ton Kloks. *Treewidth, Computations and Approximations*, volume 842 of *LNCS*. Springer, 1994.

[Lohrey and Rosowski, 2023] Markus Lohrey and Andreas Rosowski. On the complexity of diameter and related problems in permutation groups. In *ICALP'23*, pages 134:1–134:18. Dagstuhl Publishing, 2023.

[Morak, 2019] Michael Morak. Epistemic Logic Programs: A Different World View. In *ICLP'19*, pages 52–64, 2019.

[Papadimitriou and Yannakakis, 1986] Christos H. Papadimitriou and Mihalis Yannakakis. A Note on Succinct Representations of Graphs. *Inf. Control.*, 71(3):181–185, 1986.

[Papadimitriou, 1994] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Robertson and Seymour, 1985] Neil Robertson and Paul D. Seymour. Graph Minors – a Survey. In *Surveys in Combinatorics 1985*, London Mathematical Society Lecture Note Series, pages 153–171. Cambridge University Press, 1985.

[Shen and Eiter, 2016] Yi-Dong Shen and Thomas Eiter. Evaluating epistemic negation in answer set programming. *Artificial Intelligence*, 237:115–135, 2016.

[Stewart, 1991] Iain A. Stewart. Complete Problems Involving Boolean Labelled Structures and Projection Transactions. *J. Logic Comput.*, 1(6):861–882, 12 1991.

[Stockmeyer and Meyer, 1973] Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time. In *STOC'73*, pages 1–9. ACM, 1973.

[Stockmeyer, 1976] Larry J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.

[Truszczyński, 2011a] Miroslaw Truszczyński. Revisiting Epistemic Specifications. In *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning - Essays Dedicated to Michael Gelfond on the Occasion of His 65th Birthday*, LNCS, pages 315–333. Springer, 2011.

[Truszczyński, 2011b] Miroslaw Truszczyński. Trichotomy and dichotomy results on the complexity of reasoning with disjunctive logic programs. *TPLP*, 11(6):881–904, 2011.

[Valiant, 1979] Leslie G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189–201, 1979.

[Weinzierl *et al.*, 2020] Antonius Weinzierl, Richard Taupe, and Gerhard Friedrich. Advancing lazy-grounding ASP solving techniques - restarts, phase saving, heuristics, and more. *TPLP*, 20(5):609–624, 2020.

[Williams, 2008] Ryan Williams. Non-Linear Time Lower Bound for (Succinct) Quantified Boolean Formulas. *Electron. Colloquium Comput. Complex.*, TR08-076, 2008.

[Wrathall, 1976] Celia Wrathall. Complete Sets and the Polynomial-Time Hierarchy. *Theoretical Computer Science*, 3(1):23–33, 1976.