TPLP **19** (5–6): 826–840, 2019. © Cambridge University Press 2019
doi:10.1017/S1471068419000218

On Uniform Equivalence of Epistemic Logic Programs

WOLFGANG FABER and MICHAEL MORAK
University of Klagenfurt, Austria
(*e-mails:* wolfgang.faber@aau.at, michael.morak@aau.at)

STEFAN WOLTRAN
TU Wien, Vienna, Austria
(*e-mail:* woltran@dbai.tuwien.ac.at)

submitted 25 July 2019; accepted 31 July 2019

Abstract

Epistemic Logic Programs (ELPs) extend Answer Set Programming (ASP) with epistemic negation and have received renewed interest in recent years. This led to the development of new research and efficient solving systems for ELPs. In practice, ELPs are often written in a modular way, where each module interacts with other modules by accepting sets of facts as input, and passing on sets of facts as output. An interesting question then presents itself: under which conditions can such a module be replaced by another one without changing the outcome, for any set of input facts? This problem is known as uniform equivalence, and has been studied extensively for ASP. For ELPs, however, such an investigation is, as of yet, missing. In this paper, we therefore propose a characterization of uniform equivalence that can be directly applied to the language of state-of-the-art ELP solvers. We also investigate the computational complexity of deciding uniform equivalence for two ELPs, and show that it is on the third level of the polynomial hierarchy.

KEYWORDS: Logic Programming, Epistemic Negation, Non-monotonic Reasoning, Uniform Equivalence

1 Introduction

Epistemic Logic Programs (ELPs) (Gelfond 1991; Kahl et al. 2015; Shen and Eiter 2016) add epistemic operators to the language of Answer Set Programming (ASP) (Gebser et al. 2012; Brewka et al. 2011; Schaub and Woltran 2018), a generic, fully declarative logic programming language that allows for encoding problems such that the resulting answers (called *answer sets*) directly correspond to solutions of the encoded problem. In ASP, negation is usually interpreted according to the stable model semantics (Gelfond and Lifschitz 1988), that is, as negation-as-failure or default negation. Intuitively, a default negated atom $\neg a$ is true if there is no justification for a being true in the same answer set. Hence, default negation is a "local" operator defined relative to the answer set under consideration. ELPs (as defined in (Shen and Eiter 2016)) extend ASP with the epistemic negation operator **not**. An epistemically negated atom **not** a intuitively means that atom a cannot be *proven* to be true, in the sense that it is not true in every answer set. Epistemic negation is thus defined with respect to a set of answer sets, referred to as a *world view*. Deciding whether such a world view exists is Σ_P^3-complete

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

in general (Shen and Eiter 2016), whereas deciding answer set existence for ASP can be done in Σ_P^2 (Eiter and Gottlob 1995), one level lower on the polynomial hierarchy.

Michael Gelfond (1991; 1994) recognized epistemic negation to be a useful construct for ASP early on and proceeded to introduce the modal operators **K** ("known" or "provably true") and **M** ("possible" or "not provably false") to add this feature to the language. $\mathbf{K}a$ and $\mathbf{M}a$ correspond to $\neg\mathbf{not}\, a$ and $\mathbf{not}\, \neg a$, respectively. Renewed interest in recent years has revealed several flaws in the original semantics, and various new approaches (cf. e.g. (Gelfond 2011; Truszczynski 2011; Kahl 2014; Fariñas del Cerro et al. 2015; Shen and Eiter 2016)) were proposed. Also, several efficient and practical ELP solving systems have been and continue to be developed (Kahl et al. 2015; Son et al. 2017; Bichler et al. 2018).

An interesting question in the context of ELPs is when two programs are equivalent. Standard (sometimes also called ordinary or classical) equivalence is simply defined as two programs having the same world views. However, as for ASP but unlike for classical logic, this notion does not capture replaceability of ELPs. In order to be able to capture when a program is replaceable by another one, the context in which the replacement is done has to be taken into account. The notion of strong equivalence captures replaceability in any context. Strong equivalence is a well-studied topic in ASP (there the context can be any set of ASP rules) with several useful applications (Lifschitz et al. 2001; Turner 2003; Cabalar et al. 2007; Lin and Chen 2007; Eiter et al. 2013). An analogous notion has also been studied more recently for ELPs (in that case, the context can be any set of ELP rules) (Wang and Zhang 2005; Fariñas del Cerro et al. 2015; Faber et al. 2019).

For some applications, other notions of equivalence between these two extremes (no context for standard equivalence, any context for strong equivalence) are desirable. The most prominent of these intermediate equivalences is uniform equivalence, where the context is restricted to facts. This notion had been proposed for Datalog originally (Sagiv 1988; Maher 1988) and has been studied for ASP quite extensively as well (Eiter and Fink 2003; Eiter et al. 2004; Eiter et al. 2005). The notion of uniform equivalence is of course most suitable when considering a full program that is applied to various scenarios, which are represented as factual knowledge, or a program that serves as a fixed knowledge base for an agent, which then uses it together with percepts represented as facts. The concept is also useful for modular ASP programs (Lifschitz and Turner 1994; Oikarinen 2007; Janhunen et al. 2009), where sub-programs interact with each other by accepting a set of input facts and returning a set of output facts.

While uniform equivalence has been widely studied for ASP, such an investigation is, to the best of the authors' knowledge, still lacking for ELPs.

Example 1

From (Faber et al. 2019), we take the example of formulating the well-known Closed World Assumption via ELP rules. Two formulations of the CWA have been proposed in this context. In (Gelfond 1991), a rule for CWA is proposed that, in the language of ELPs, can be formulated as

$$p' \leftarrow \neg\mathbf{not}\, \neg p.$$

Intuitively, this says that p' (meaning the negation of p) shall be true if there is no possible world where p is true. In (Shen and Eiter 2016), a different rule is proposed:

$$p' \leftarrow \mathbf{not}\, p.$$

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

Here, intuitively, p' shall be true if there is a possible world where p is false. While the two formulations are equivalent in that they share the same world views, in (Faber et al. 2019), it was shown that they are, however, not strongly equivalent. However, this does not tell us anything about the uniform equivalence of these two rules.

In order to analyze cases like the one presented in Example 1 above, it is the aim of this paper to study uniform equivalence for ELPs, that is, the question of whether, given two ELPs Π_1 and Π_2, for any set of facts D, the combined programs $\Pi_1 \cup D$ and $\Pi_2 \cup D$ have the same world views. According to (Faber et al. 2019), two versions of (ordinary) equivalence between ELPs can be defined: one where all candidate world views are equal, and one where only the world views (that is, candidate world views that minimize the number of assumptions) are equal. In (Faber et al. 2019), two versions of strong equivalence, relative to these ordinary equivalence notions, are defined, and then subsequently shown to coincide. We will follow the same approach. Interestingly, we will see that for uniform equivalence, the two versions do not coincide.

Contributions. The main contributions of this paper are the following:

- We formally define two versions of uniform equivalence for ELPs (based on the input language of today's ELP solvers) that appropriately extend uniform equivalence for ASP, based on existing notions of equivalence for ELPs, as used in (Faber et al. 2019).
- We provide an analysis of the two different notions of uniform equivalence for ELPs and characterize their relationship. Furthermore, a model-theoretic characterization is offered, based on a so-called UE-function, in the same vein as the SE-function was introduced in (Faber et al. 2019) to characterize strong equivalence.
- We then show that testing uniform equivalence of two ELPs is Π_3^P-complete, that is, the complexity of this test jumps up one level on the polynomial hierarchy compared to ASP, and hence is much harder than testing strong equivalence, which for ELPs is only coNP-complete (Faber et al. 2019).

Organization. The remainder of the paper is structured as follows. Section 2 gives an overview of the relevant definitions needed in the main sections of the paper, including the language of ASP, ELPs, and the notions of strong and uniform equivalence for the former. Section 3 defines two different notions of uniform equivalence for ELPs, shows that, in contrast to strong equivalence, these notions do not coincide, and finally offers a model-theoretic characterization of these notions of uniform equivalence, called the UE-function. Following this characterization, we investigate the computational complexity of deciding uniform equivalence in Section 4. We then offer some concluding remarks in Section 5.

2 Preliminaries

Answer Set Programming (ASP). A *ground logic program* with nested negation (also called answer set program, ASP program, or, simply, logic program) is a pair $\Pi = (\mathcal{A}, \mathcal{R})$, where $\mathcal{A}$ is a set of propositional (i.e. ground) atoms and $\mathcal{R}$ is a finite set of rules of the form

$$a_1 \vee \cdots \vee a_l \leftarrow a_{l+1}, \ldots, a_m, \neg\ell_1, \ldots, \neg\ell_n; \tag{1}$$

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

where the comma symbol stands for conjunction, $0 \leq l \leq m$, $0 \leq n$, $a_i \in \mathcal{A}$ for all $1 \leq i \leq m$, and each ℓ_i is a *literal*, that is, either an atom a or its (default) negation $\neg a$ for any atom $a \in \mathcal{A}$. Note that, therefore, doubly negated atoms may occur. We will sometimes refer to such rules as *standard rules*. Each rule $r \in \mathcal{R}$ of form (1) consists of a *head* $H(r) = \{a_1, \ldots, a_l\}$ and a *body* $B(r) = \{a_{l+1}, \ldots, a_m, \neg\ell_1, \ldots, \neg\ell_n\}$. We denote the *positive* body by $B^+(r) = \{a_{l+1}, \ldots, a_m\}$. A rule where $l = 1$, $m = l$, and $n = 0$ is called a *fact*.

An *interpretation* I (over $\mathcal{A}$) is a set of atoms, that is, $I \subseteq \mathcal{A}$. A literal ℓ is true in an interpretation $I \subseteq \mathcal{A}$, denoted $I \vDash \ell$, if $a \in I$ and $\ell = a$, or if $a \notin I$ and $\ell = \neg a$; otherwise ℓ is false in I, denoted $I \nvDash \ell$. Finally, for some literal ℓ, we define that $I \vDash \neg\ell$ if $I \nvDash \ell$. This notation naturally extends to sets of literals. An interpretation M is called a *model* of r, denoted $M \vDash r$, if, whenever $M \vDash B(r)$, it holds that $M \vDash H(r)$. We denote the set of models of r by $mods(r)$; the models of a logic program $\Pi = (\mathcal{A}, \mathcal{R})$ are given by $mods(\Pi) = \bigcap_{r \in \mathcal{R}} mods(r)$. We also write $I \vDash r$ (resp. $I \vDash \Pi$) if $I \in mods(r)$ (resp. $I \in mods(\Pi)$).

The GL-reduct Π^I of a logic program $\Pi = (\mathcal{A}, \mathcal{R})$ with respect to an interpretation I is the program $(\mathcal{A}, \mathcal{R}^I)$, where $\mathcal{R}^I = \{H(r) \leftarrow B^+(r) \mid r \in \mathcal{R}, \forall \neg\ell \in B(r) : I \vDash \neg\ell\}$.

Definition 2
(Gelfond and Lifschitz 1988; Gelfond and Lifschitz 1991; Lifschitz et al. 1999) $M \subseteq \mathcal{A}$ is an *answer set* of a logic program Π if (1) $M \in mods(\Pi)$ and (2) there is no subset $M' \subset M$ such that $M' \in mods(\Pi^M)$.

The set of answer sets of a logic program Π is denoted by $AS(\Pi)$. The *consistency problem* of ASP, that is, to decide whether for a given logic program Π it holds that $AS(\Pi) \neq \emptyset$, is Σ_P^2-complete (Eiter and Gottlob 1995), and remains so also in the case where doubly negated atoms are allowed in rule bodies (Pearce et al. 2009).

Strong and Uniform Equivalence for Logic Programs. Two logic programs $\Pi_1 = (\mathcal{A}, \mathcal{R}_1)$ and $\Pi_2 = (\mathcal{A}, \mathcal{R}_2)$ are *equivalent* iff they have the same set of answer sets, that is, $AS(\Pi_1) = AS(\Pi_2)$. The two logic programs are *strongly equivalent* iff for any third logic program $\Pi = (\mathcal{A}, \mathcal{R})$ it holds that the combined logic program $\Pi_1 \cup \Pi = (\mathcal{A}, \mathcal{R}_1 \cup \mathcal{R})$ is equivalent to the combined logic program $\Pi_2 \cup \Pi = (\mathcal{A}, \mathcal{R}_2 \cup \mathcal{R})$. They are *uniformly equivalent* iff they are strongly equivalent for any third program Π consisting only of facts. An *SE-model* (Turner 2003) of a logic program $\Pi = (\mathcal{A}, \mathcal{R})$ is a tuple of interpretations (X, Y), where $X \subseteq Y \subseteq \mathcal{A}$, $Y \vDash \Pi$, and $X \vDash \Pi^Y$. The set of SE-models of a logic program Π is denoted $\mathcal{SE}(\Pi)$. Note that for every model Y of Π, (Y, Y) is an SE-model of Π, since $Y \vDash \Pi$ implies that $Y \vDash \Pi^Y$. An SE-model (X, Y) of Π is a *UE-model* of Π (Eiter and Fink 2003) iff either $X = Y$, or $X \subset Y$ and there is no other SE-model $(X', Y) \in \mathcal{SE}(\Pi)$ such that $X \subset X' \subset Y$. The set of UE-models of Π is denoted $\mathcal{UE}(\Pi)$. Hence, UE-models are precisely those SE-models, where the X component is either Y, or subset-maximal w.r.t. the other SE-models.

Two logic programs (over the same set of atoms) are uniformly equivalent iff they have the same UE-models and checking uniform equivalence is Π_2^P-complete in general (Eiter and Fink 2003).

Epistemic Logic Programs. An *epistemic literal* is a formula $\mathbf{not}\,\ell$, where ℓ is a literal and **not** is the epistemic negation operator. A *ground epistemic logic program (ELP)* is

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

a triple $\Pi = (\mathcal{A}, \mathcal{E}, \mathcal{R})$, where $\mathcal{A}$ is a set of propositional atoms, $\mathcal{E}$ is a set of epistemic literals over the atoms $\mathcal{A}$, and $\mathcal{R}$ is a finite set of *ELP rules*, which are

$$a_1 \vee \cdots \vee a_k \leftarrow \ell_1, \ldots, \ell_m, \xi_1, \ldots, \xi_j, \neg\xi_{j+1}, \ldots, \neg\xi_n,$$

where each $a_i \in \mathcal{A}$ is an atom, each ℓ_i is a literal, and each $\xi_i \in \mathcal{E}$ is an epistemic literal. Note that usually $\mathcal{E}$ is defined implicitly to be the set of all epistemic literals appearing in the rules $\mathcal{R}$; however, making the domain of epistemic literals explicit will prove useful for our purposes. The *union* of two ELPs $\Pi_1 = (\mathcal{A}_1, \mathcal{E}_1, \mathcal{R}_1)$ and $\Pi_2 = (\mathcal{A}_2, \mathcal{E}_2, \mathcal{R}_2)$ is the ELP $\Pi_1 \cup \Pi_2 = (\mathcal{A}_1 \cup \mathcal{A}_2, \mathcal{E}_1 \cup \mathcal{E}_2, \mathcal{R}_1 \cup \mathcal{R}_2)$.

For a set $\mathcal{E}$ of epistemic literals, a subset $\Phi \subseteq \mathcal{E}$ of epistemic literals is called an *epistemic guess* (or, simply, a *guess*). The following definition provides a way to check whether a set of interpretations is compatible with a guess Φ.

Definition 3
Let $\mathcal{A}$ be a set of atoms, $\mathcal{E}$ be a set of epistemic literals over $\mathcal{A}$, and $\Phi \subseteq \mathcal{E}$ be an epistemic guess. A set $\mathcal{I}$ of interpretations over $\mathcal{A}$ is called Φ-*compatible w.r.t.* $\mathcal{E}$, iff

1. $\mathcal{I} \neq \emptyset$;
2. for each epistemic literal $\mathbf{not}\,\ell \in \Phi$, there exists an interpretation $I \in \mathcal{I}$ such that $I \nvDash \ell$; and
3. for each epistemic literal $\mathbf{not}\,\ell \in \mathcal{E} \setminus \Phi$, for all interpretations $I \in \mathcal{I}$ it holds that $I \vDash \ell$.

For an ELP $\Pi = (\mathcal{A}, \mathcal{E}, \mathcal{R})$, the *epistemic reduct* (Shen and Eiter 2016) of the program Π w.r.t. a guess Φ, denoted Π^Φ, consists of the rules $\mathcal{R}^\Phi = \{r^\neg \mid r \in \mathcal{R}\}$, where $r^\neg$ is defined as the rule $r \in \mathcal{R}$ where all occurrences of epistemic literals $\mathbf{not}\,\ell \in \Phi$ are replaced by $\top$, and all remaining epistemic negation symbols **not** are replaced by default negation $\neg$. Note that, after this transformation, $\Pi^\Phi = (\mathcal{A}, \mathcal{R}^\Phi)$ is a logic program without epistemic negation[1]. This leads to the following, central definition.

Definition 4
Let $\Pi = (\mathcal{A}, \mathcal{E}, \mathcal{R})$ be an ELP. A set $\mathcal{M}$ of interpretations over $\mathcal{A}$ is a *candidate world view (CWV)* of Π if there is an epistemic guess $\Phi \subseteq \mathcal{E}$ such that $\mathcal{M} = AS(\Pi^\Phi)$ and $\mathcal{M}$ is compatible with Φ w.r.t. $\mathcal{E}$. The set of all CWVs of an ELP Π is denoted by $cwv(\Pi)$.

Let us consider an example for illustrative purposes.

Example 5
Let $\mathcal{A} = \{p, p'\}$, $\mathcal{E} = \{\mathbf{not}\,\neg p\}$, $\Pi = (\mathcal{A}, \mathcal{E}, \mathcal{R})$ with $\mathcal{R}$ containing only rule $p' \leftarrow \neg\mathbf{not}\,\neg p$, a well-known formulation of the closed world assumption proposed in (Gelfond 1991)[2].

We obtain $cwv(\Pi) = \{\{\{p'\}\}\}$ as guess $\Phi = \emptyset$ is compatible with $AS(\Pi^\Phi = \{p' \leftarrow \neg p\}) = \{\{p'\}\}$, while no other guesses are compatible with the answer sets of the respective epistemic reducts.

[1] In fact, Π^Φ may contain triple-negated atoms $\neg\neg\neg a$. But, according to the definition of the GL-reduct in (Lifschitz et al. 1999), such formulas are equivalent to simple negated atoms $\neg a$, and we treat them as such.

[2] In fact, in (Gelfond 1991), the author proposes the rule $\sim p \leftarrow \neg\mathbf{M}p$, where $\sim$ is a third kind of negation, usually referred to as strong negation, not considered in this paper. It can be simulated by replacing occurrences of $\sim p$ by a fresh atom p' and adding a constraint rule $\leftarrow p, p'$ that excludes p and p' to hold simultaneously.

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

Following the principle of knowledge minimization, a *world view*, in (Shen and Eiter 2016), is defined as follows.

Definition 6
Let $\Pi = (\mathcal{A}, \mathcal{E}, \mathcal{R})$ be an ELP. $\mathcal{C} \in cwv(\Pi)$ is called *world view (WV)* of Π if its associated guess Φ is subset-maximal, i.e. there is no $\mathcal{C}' \in cwv(\Pi)$ with associated guess $\Phi' \supset \Phi$.

Note that in Example 5 there is only one CWV per program; hence the associated guesses are subset-maximal, and the sets of CWVs and WVs coincide.

Note that given two ELPs $\Pi_1 = (\mathcal{A}_1, \mathcal{E}_1, \mathcal{R}_1)$ and $\Pi_2 = (\mathcal{A}_2, \mathcal{E}_2, \mathcal{R}_2)$, we can always assume that $\mathcal{A}_1 = \mathcal{A}_2$ and $\mathcal{E}_1 = \mathcal{E}_2$ without changing the (candidate) world views of the two programs (Faber et al. 2019). In order to simplify our investigation, we will make use of this assumption when we compare two ELPs.

One of the main reasoning tasks regarding ELPs is the *world view existence problem*, that is, given an ELP Π, decide whether a WV (or, equivalently, CWV) exists. This problem is Σ_3^P-complete (Shen and Eiter 2016).

3 Uniform Equivalence for ELPs

In this section, we will investigate the uniform equivalence of ELPs, in particular, focusing on how to extend this concept (Eiter and Fink 2003) from logic programs to ELPs. In order to begin this investigation, we will first define (ordinary) equivalence of two ELPs. The following definition was recently proposed in (Faber et al. 2019).

Definition 7
Two ELPs are *WV-equivalent* (resp. *CWV-equivalent*) iff their world views (resp. candidate world views) coincide.

Note that CWV-equivalence immediately implies WV-equivalence.

We now continue by defining uniform equivalence for ELPs. One motivation for such a kind of equivalence is module optimization: we would like to replace a module in an ELP, that accepts a set of input facts and provides a set of output facts, with another (hopefully more efficient) formulation without changing the semantics (i.e. WVs or CWVs). Based on the two equivalence notions defined above and using ideas from work done in the area of logic programs (Eiter and Fink 2003), we propose two notions of uniform equivalence for ELPs.

Definition 8
Let Π_1 and Π_2 be two ELPs. Π_1 and Π_2 are

- *uniformly WV-equivalent* iff, for every set of ground facts D, $\Pi_1 \cup D$ and $\Pi_2 \cup D$ are WV-equivalent; and
- *uniformly CWV-equivalent* iff, for every set of ground facts D, $\Pi_1 \cup D$ and $\Pi_2 \cup D$ are CWV-equivalent.

One could be tempted to define uniform equivalence for ELPs simply in terms of the UE-models (Eiter and Fink 2003) of the epistemic reducts, for each possible epistemic guess. However, this approach does not capture our intentions, as the following example shows:

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

Example 9
Take the two ELPs Π_1 and Π_2

$$\Pi_1 = (\mathcal{A}, \mathcal{E}, \mathcal{R}_1) \qquad \Pi_2 = (\mathcal{A}, \mathcal{E}, \mathcal{R}_2)$$
$$\mathcal{R}_1 = \{p \leftarrow \mathbf{not}\, p\} \qquad \mathcal{R}_2 = \{p \leftarrow \neg p\}$$

with $\mathcal{A} = \{p\}$ and $\mathcal{E} = \{\mathbf{not}\, p\}$. Now, for the guess $\Phi_1 = \emptyset$, note that $\Pi_1^{\Phi_1} = \Pi_2^{\Phi_1}$ and hence, trivially, the UE-models are also the same. However, for the guess $\Phi_2 = \mathcal{E}$, $\Pi_1^{\Phi_2}$ consists of the rule $p \leftarrow \top$, while $\Pi_2^{\Phi_2}$ reduces to $p \leftarrow \neg p$. It can be checked that the UE-models of these two epistemic reducts w.r.t. Φ_2 are not the same and are hence not uniformly equivalent in the sense of (Eiter and Fink 2003). However, it turns out that the guess Φ_2 can never give rise to a CWV, since it requires that there is an answer set not containing p, but both $\Pi_1^{\Phi_2}$ and $\Pi_2^{\Phi_2}$ require that p is true in all answer sets of the CWV.

The example above implies that, when establishing uniform equivalence for ELPs, we need a more involved construction. Before we turn to the subject of the characterization, however, we will first investigate the relationship between uniform CWV and WV-equivalence.

Clearly, it holds that uniform CWV-equivalence is the stronger notion, as it directly implies uniform WV-equivalence. It can be shown that this relationship is strict, and hence the two notions are actually distinct, as the following proposition states:

Proposition 10
Let Π_1 and Π_2 be two ELPs. It holds that

1. when Π_1 and Π_2 are uniformly CWV-equivalent, then they are uniformly WV-equivalent; and
2. the ELPs Π_1 and Π_2 may be such that Π_1 and Π_2 are uniformly WV-equivalent but not uniformly CWV-equivalent.

Proof
(1) As observed after Definition 7, if two ELPs are CWV-equivalent then they are WV-equivalent. This holds, in particular, for any set of facts D, and the ELPs $\Pi_1 \cup D$ and $\Pi_2 \cup D$.

(2) We will prove this by example. Take the ELPs $\Pi_1 = (\mathcal{A}, \mathcal{E}, \mathcal{R}_1)$ and $\Pi_2 = (\mathcal{A}, \mathcal{E}, \mathcal{R}_2)$ built as follows. Let $\mathcal{R}$ be the following set of rules:

$$\begin{aligned}\mathcal{R} = \{&a \vee b \leftarrow \mathbf{not}\, \neg a, \mathbf{not}\, \neg b;\\ &a \leftarrow b;\\ &b \leftarrow a;\\ &c \leftarrow d, \neg a, \neg b;\\ &d \leftarrow c, \neg a, \neg b\}\end{aligned}$$

Now, let

$$\mathcal{R}_1 = \mathcal{R} \cup \{c \vee d \leftarrow \neg a, \neg b\}$$
$$\mathcal{R}_2 = \mathcal{R} \cup \{c \leftarrow \neg d, \neg a, \neg b;\ d \leftarrow \neg c, \neg a, \neg b\}.$$

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

We will show that Π_1 and Π_2 are uniformly WV-equivalent, but not uniformly CWV-equivalent.

Note that $\mathcal{A} = \{a, b, c, d\}$ and $\mathcal{E} = \{\mathbf{not}\, \neg a, \mathbf{not}\, \neg b\}$. To prove our claim, let us first examine the first three rules of $\mathcal{R}$. From these rules, it is not difficult to check that there are two epistemic guesses that lead to CWVs, namely $\Phi_1 = \emptyset$ and $\Phi_2 = \mathcal{E}$. The CWV for Φ_2 is the set $\{\{a, b\}\}$. Note that Φ_2 is subset-maximal, and hence this set is also a WV. Note further that adding any set of facts $D \subseteq \mathcal{A}$ to Π_1 or Π_2 will simply change the WV to $\{\{a, b\} \cup D\}$, which is still a valid WV w.r.t. guess Φ_2 for both ELPs. However, the CWVs w.r.t. guess Φ_1 differ already for $D = \emptyset$: for $\Pi_1 \cup D$ it is $\{\{a, b, c, d\}\}$, whereas for $\Pi_2 \cup D$ no CWV exists. Hence, we have that Π_1 and Π_2 are uniformly WV-equivalent, but not uniformly CWV-equivalent, as desired. □

The above result shows an interesting distinction between uniform equivalence and strong equivalence when regarding ELPs. As shown in (Faber et al. 2019), the different notions of strong equivalence considered therein coincide (that is, regarding strong equivalence w.r.t. WVs or CWVs does not make a difference), this is not the case for uniform equivalence, where there is an actual distinction between uniform CWV- and uniform WV-equivalence.

A further observation that can be made is that both forms of uniform equivalence for ELPs strictly generalize the notion of uniform equivalence for ASP, as the following result shows.

Theorem 11
Let $\Pi_1 = (\mathcal{A}, \mathcal{R}_1)$ and $\Pi_2 = (\mathcal{A}, \mathcal{R}_2)$ be two logic programs, and $\Pi_1' = (\mathcal{A}, \mathcal{E}, \mathcal{R}_1)$ and $\Pi_2' = (\mathcal{A}, \mathcal{E}, \mathcal{R}_2)$ be two ELPs containing the same rules, respectively, and where $\mathcal{E} = \emptyset$. Then, the following three statements are equivalent:

1. Π_1 and Π_2 are uniformly equivalent,
2. Π_1' and Π_2' are uniformly CWV-equivalent, and
3. Π_1' and Π_2' are uniformly WV-equivalent.

Proof
Note that, since $\mathcal{E} = \emptyset$, both ELPs Π_1' and Π_2' have at most one CWV (and hence WV), namely the set $AS(\Pi_1)$ and $AS(\Pi_2)$, respectively, in case these sets are non-empty. Otherwise, if Π_1 or Π_2 is inconsistent, then Π_1' or Π_2' do not have any CWVs, respectively.

(2) $\Leftrightarrow$ *(3).* Since Π_1' and Π_2' have at most one CWV that corresponds to the guess $\Phi = \mathcal{E} = \emptyset$, the notions of uniform WV-equivalence and uniform CWV-equivalence coincide.

(1) $\Rightarrow$ *(2).* By assumption, Π_1 and Π_2 are uniformly equivalent. Towards a contradiction, assume that there is a set of facts $D \subseteq \mathcal{A}$, such that $\Pi_1' \cup D$ is not CWV-equivalent to $\Pi_2' \cup D$. Note that, since the only epistemic guess possible is the guess $\Phi = \mathcal{E} = \emptyset$, any non-empty set of answer sets satisfies Φ. Further, note that for $i \in \{1, 2\}$ it holds that $(\Pi_i' \cup D)^\Phi = \Pi_i'^\Phi \cup D = \Pi_i \cup D$. Hence, we have that $AS(\Pi_1 \cup D) \neq AS(\Pi_2 \cup D)$, contradicting our assumption.

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

(2) ⇒ (1). This follows from a similar argument as the one above. □

Having defined the notions of uniform equivalence for ELPs, we aim to characterize it in a similar fashion as was done for strong equivalence for ELPs in (Faber et al. 2019), and for logic programs in (Turner 2003). Unfortunately, it seems that an "interesting" characterization, as in these two papers, is not possible for uniform equivalence of ELPs. Due to the complex interactions between epistemic guesses, answer sets, and the sets of facts added, only a very straightforward characterization of uniform equivalence for ELPs is possible. We formulate this as a so-called UE-function, in the spirit of the SE-function for strong equivalence as given in (Faber et al. 2019).

Definition 12
Let $\Pi = (\mathcal{A}, \mathcal{E}, \mathcal{R})$ be an ELP, and let $W \in \{\mathrm{CWV}, \mathrm{WV}\}$. Then, $\mathcal{UE}_\Pi^W : 2^\mathcal{E} \times 2^\mathcal{A} \to 2^\mathcal{A}$ is called the W-UE-function of Π iff, for any epistemic guess $\Phi \subseteq \mathcal{E}$ and any set of facts $D \subseteq \mathcal{A}$, it holds that

$$\mathcal{UE}_\Pi^W(\Phi, D) = \begin{cases} \mathcal{M} & \text{if } \mathcal{M} \text{ is a CWV of type } W \text{ of } \Pi \text{ w.r.t. } \Phi \\ \emptyset & \text{otherwise,} \end{cases}$$

where $\mathcal{M} = AS((\Pi \cup D)^\Phi)$.

As can be seen, the characterization is rather straightforward (for a given ELP Π, it maps an epistemic guess and a set of facts to the CWV or WV that arises w.r.t. the guess when adding the set of facts to Π). The following result follows immediately from the construction of the UE-function:

Theorem 13
For $W \in \{\mathrm{CWV}, \mathrm{WV}\}$, two ELPs Π_1 and Π_2 are W-equivalent iff their W-UE-functions coincide.

While this characterization thus is far less interesting than the characterization for strong equivalence, it seems that the multiple layers involved in computing world views of ELPs make a more interesting construction, which tries to directly use UE-models from classical logic programming (Eiter and Fink 2003), impossible. Further evidence of this will be presented in the next section, where we investigate the computational complexity of deciding uniform equivalence. While deciding strong equivalence for ELPs is coNP-complete, we will see that the same task for uniform equivalence is much harder, making it unlikely that an elegant compact representation, like the UE-models proposed in (Eiter and Fink 2003), or the SE-function from (Faber et al. 2019) can be found.

Before turning our attention to this topic, however, let us briefly return to our example from Section 1. Recall that in Example 1, we have seen two versions to formulate the CWA using ELPs. We shall investigate these two formulations w.r.t. their uniform equivalence.

Example 14
It turns out that the two formulations of CWA shown in Example 1 are in fact both uniformly CWV-equivalent and uniformly WV-equivalent ELPs. This can be verified by constructing the relevant UE-functions. Let $\Pi_{\mathrm{Gelfond}} = (\mathcal{A}, \mathcal{E}, \mathcal{R}_{\mathrm{Gelfond}})$ be the formulation of the CWA from (Gelfond 1991), and $\Pi_{\mathrm{ShenEiter}} = (\mathcal{A}, \mathcal{E}, \mathcal{R}_{\mathrm{ShenEiter}})$ be the formulation from (Shen and Eiter 2016), where $\mathcal{A} = \{p', p\}$ and $\mathcal{E} = \{\mathbf{not}\, p, \mathbf{not}\, \neg p\}$. Omitting all combinations of epistemic guesses and sets of facts where the UE-functions

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

return $\emptyset$, the UE-functions, for both $W \in \{\mathrm{WV}, \mathrm{CWV}\}$, look as follows:

$$\begin{aligned}\mathcal{UE}^{W}_{\Pi_{\mathrm{Gelfond}}} = \mathcal{UE}^{W}_{\Pi_{\mathrm{ShenEiter}}} = \{ & \\ & (\{\mathbf{not}\, p\}, \emptyset, \{\{p'\}\}), \\ & (\{\mathbf{not}\, p\}, \{p'\}, \{\{p'\}\}), \\ & (\{\mathbf{not}\, \neg p\}, \{p\}, \{\{p', p\}\}), \\ & (\{\mathbf{not}\, \neg p\}, \{p', p\}, \{\{p', p\}\}) \\ \} & \end{aligned}$$

From this, since the UE-functions for the two formulations coincide, we observe that in the context of uniform equivalence, these two formulations are, for all intents and purposes, interchangeble.

With the above example, we are able to formally establish the following result, comparing our notions of uniform equivalence for ELPs to established notions of equivalence. We observe that, as expected, uniform equivalence is strictly stronger than (ordinary) equivalence, but strictly weaker than strong equivalence.

Theorem 15
For ELPs, it holds that

1. strong equivalence is strictly stronger than uniform CWV-equivalence;
2. uniform CWV-equivalence is strictly stronger than uniform WV-equivalence;
3. uniform CWV-equivalence is strictly stronger than (ordinary) CWV-equivalence;
4. uniform WV-equivalence is strictly stronger than (ordinary) WV-equivalence.

Proof
Several observations follow trivially from the relevant definitions: strong equivalence implies uniform CWV-equivalence (since sets of atoms are also ELPs), which clearly implies uniform WV-equivalence (since WVs are a subset of CWVs). Finally, since uniform CWV-equivalence (resp. WV-equivalence) require that the CWVs (resp. WVs) are the same for any set of added atoms—in particular, the empty set of atoms—they directly imply (ordinary) CWV-equivalence (resp. WV-equivalence).

To establish that the separations between these equivalence notions are indeed strict, we make use of several separating examples. Statement (1) is shown by Examples 1 and 14, which exhibit two ELPs that are uniformly CWV-equivalent, but not strongly equivalent. Statement (2) follows from statement (2) of Proposition 10. Finally, statements (3) and (4) follow from the fact that uniform CWV-equivalence and uniform WV-equivalence both generalize uniform equivalence for ground logic programs (cf. Theorem 11), and, in this case, uniform equivalence is already strictly stronger than ordinary equivalence; see e.g. (Eiter and Fink 2003, Example 10). □

4 Complexity of ELP Uniform Equivalence

Having defined our characterization of uniform equivalence for ELPs, in this section, we will now focus on the question of the computational complexity of deciding whether two ELPs are uniformly equivalent. It turns out that this task is of similar hardness

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

as deciding the CWV existence problem for ELPs, that is, on the third level of the polynomial hierarchy. Hence, it is one level higher in the polynomial hierarchy than for plain ground (disjunctive) logic programs under the stable model semantics, for which uniform-equivalence checking is Π_2^P-complete (Eiter and Fink 2003, Theorem 10). The following result states this formally:

Theorem 16
Deciding uniform CWV-equivalence of two ELPs is Π_3^P-complete.

Proof
For this proof, assume that $\Pi_1 = (\mathcal{A}, \mathcal{E}, \mathcal{R}_1)$ and $\Pi_2 = (\mathcal{A}, \mathcal{E}, \mathcal{R}_2)$ are two ELPs (w.l.o.g. over the same set of atoms and epistemic literals).

Upper Bound. As stated in (Shen and Eiter 2016, Theorem 4), given an epistemic guess Φ and an ELP Π, verifying that Π has a CWV w.r.t. Φ can be done in D_2^P, and hence via two calls to a Σ_2^P oracle. We therefore obtain a straightforward guess-and-check algorithm that runs in non-deterministic polynomial time with a Σ_2^P oracle, checks non-uniform equivalence between two ELPs, and works as follows: guess a set of atoms $D \subseteq \mathcal{A}$, an epistemic guess $\Phi \subseteq \mathcal{E}$, and a set of facts $M \subseteq \mathcal{A}$. Then, use a Σ_2^P-oracle to check that one of the following two conditions hold: (i) Φ leads to a CWV for $\Pi_1 \cup D$, but not for $\Pi_2 \cup D$, or (ii) Φ leads to a CWV for both $\Pi_1 \cup D$ and $\Pi_2 \cup D$, but that M is an answer set that exists only in exactly one of these two CWVs.

Lower Bound. We will show Π_3^P-hardness via reduction from 3-QBF solving. We will construct two ELPs Π_1 and Π_2 such that they are uniformly equivalent iff a given 3-QBF is unsatisfiable. To this end, we will make use of the reduction from 3-QBF solving to CWV existence offered in (Shen and Eiter 2016, Proof of Theorem 5), on which our reduction is based. Let $\exists \mathbf{X} \forall \mathbf{Y} \exists \mathbf{Z}\ \Psi(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ be a 3-QBF formula in conjunctive normal form, where each clause has the form $\ell_1 \vee \ell_2 \vee \ell_3$, where each ℓ is a literal over the variables in $\mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z}$. In (Shen and Eiter 2016), it is assumed w.l.o.g. that the 3-QBF evaluates to true whenever all variables in $\mathbf{Y}$ are replaced by $\top$. This does not change the hardness of the problem, and we make use of the same assumption. For a 3-QBF formula as above, we construct the ELP $\Pi_1 = (\mathcal{A}, \mathcal{E}, \mathcal{R}_1)$ over the atoms $\mathcal{A} = \{w, \overline{w} \mid w \in \mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z}\} \cup \{\mathit{false}, \mathit{sat}\}$ using the well-known technique of saturation (Eiter and Gottlob 1995). Π_1 contains the following set of rules, where ℓ^* converts a literal a into atom $\overline{a}$ and literal $\neg a$ into atom a:

- for each $x \in \mathbf{X}$:

$$x \leftarrow \mathbf{not}\, \overline{x}, \tag{2}$$

$$\overline{x} \leftarrow \mathbf{not}\, x; \tag{3}$$

- for each $y \in \mathbf{Y}$:

$$y \leftarrow \neg \overline{y}, \tag{4}$$

$$\overline{y} \leftarrow \neg y, \tag{5}$$

$$\bot \leftarrow \neg \mathbf{not}\, y, \tag{6}$$

$$\bot \leftarrow \neg \mathbf{not}\, \overline{y}; \tag{7}$$

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

- for each $z \in \mathbf{Z}$:

$$z \vee \overline{z}, \tag{8}$$

$$z \leftarrow \mathit{sat}, \tag{9}$$

$$\overline{z} \leftarrow \mathit{sat}; \tag{10}$$

- for each clause $\ell_1 \vee \ell_2 \vee \ell_3$ in Ψ:

$$\mathit{sat} \leftarrow \ell_1^*, \ell_2^*, \ell_3^*; \tag{11}$$

- and the two rules

$$\mathit{false} \leftarrow \mathbf{not}\, \mathit{false}, \mathbf{not}\, \neg \mathit{sat}, \tag{12}$$

$$\bot \leftarrow \neg \mathbf{not}\, \mathit{false}. \tag{13}$$

The construction of $\Pi_2 = (\mathcal{A}, \mathcal{E}, \mathcal{R}_2)$ differs from Π_1 in only one respect: for Π_2, ℓ^* converts literals into $\top$. Note that, therefore, Π_2 contains the fact *sat*. This completes the main part of our construction. Let us now explore how our construction works. From (Shen and Eiter 2016), we have that program Π_1 has a CWV iff the 3-QBF $\exists\mathbf{X}\forall\mathbf{Y}\exists\mathbf{Z}\Psi$ is satisfiable, and hence, conversely, Π_1 has no CWVs iff the QBF is unsatisfiable (since, in this case, in the GL-reduct, the atom *sat* is always derived, and hence, constraint (12) destroys any potential CWV). Note that any CWV $\mathcal{M}$ for Π_1 has the following structure: a guess Φ leading to $\mathcal{M}$ will contain a subset of $\{\mathbf{not}\, x, \mathbf{not}\, \overline{x} \mid x \in \mathbf{X}\}$ representing an assignment on the variables of $\mathbf{X}$. Further, $\Phi \supseteq \{\mathbf{not}\, y, \mathbf{not}\, \overline{y} \mid y \in \mathbf{Y}\}$, since each answer set in the CWV represents precisely one assignment on the variables $\mathbf{Y}$, and all possible such assignments must appear in the CWV. Finally, $\mathbf{not}\, \mathit{false} \in \Phi$ and $\mathbf{not}\, \neg \mathit{sat} \notin \Phi$, via constraint (12). For the precise reasoning behind this construction, please see (Shen and Eiter 2016, Proof of Theorem 5).

Towards our goal, we must show two things: (a) in cases where the 3-QBF $\exists\mathbf{X}\forall\mathbf{Y}\exists\mathbf{Z}\Psi$ is satisfiable, there exists a set of atoms $D \subseteq \mathcal{A}$, such that $\Pi_1 \cup D$ and $\Pi_2 \cup D$ are not equivalent (i.e. have differing CWVs), and hence, Π_1 and Π_2 are not uniformly equivalent; and (b) in cases where the 3-QBF is unsatisfiable for all sets of facts $D \subseteq \mathcal{A}$ it holds that $\Pi_1 \cup D$ is equivalent to $\Pi_2 \cup D$ (i.e. they have the same CWVs), and hence, Π_1 and Π_2 are uniformly equivalent.

Showing (a) is straightforward: simply take $D = \emptyset$. Π_1 has a CWV (via correctness of the reduction in (Shen and Eiter 2016) as explained above), whereas Π_2 does not have a CWV, since the atom *sat* is always derived in any GL-reduct of Π_2, destroying each potential CWV. Showing (b) is a little more involved. We will show this by contradiction. To this end, assume that the 3-QBF is unsatisfiable, but some set of facts $D \subseteq \mathcal{A}$ exists, such that $\Pi_1 \cup D$ and $\Pi_2 \cup D$ are not equivalent. Our plan is to show that D cannot contain any atoms from $\mathcal{A}$, but also cannot be empty. Let us look at the atoms in $\mathcal{A}$ in turn.

sat $\in D$**:** in this case, the only difference between Π_1 and Π_2, namely rules of the form (11), disappears, and hence, $\Pi_1 \cup D$ and $\Pi_2 \cup D$ cannot have differing CWVs; a contradiction.

false $\in D$**:** in this case, the atom *false* is true in every answer set, regardless of the epistemic guess Φ, in both $\Pi_1 \cup D$ and $\Pi_2 \cup D$. Hence, $\mathbf{not}\, \mathit{false} \notin \Phi$. But then,

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

rule (13) becomes $\bot \leftarrow \neg\neg \mathit{false}$ in the epistemic reduct w.r.t. Φ, and no answer set can both contain D and satisfy this constraint; a contradiction.

$\{y, \overline{y} \mid y \in \mathbf{Y}\} \cap D \neq \emptyset$**:** this case is similar to the case of *false*. If any such atom y or $\overline{y}$ is in D, and hence true in every answer set of any epistemic reduct, then constraints (6) and (7) will prevent that epistemic reduct from having any answer sets for both Π_1 and Π_2; a contradiction.

$D \subseteq \{w, \overline{w} \mid w \in \mathbf{X} \cup \mathbf{Z}\}$**:** since the 3-QBF is unsatisfiable, we know that for any assignment on the variables in $\mathbf{X}$ and $\mathbf{Z}$ there is an assignment on the variables in $\mathbf{Y}$ such that Ψ is false. Since, from the last paragraph, we know that $D \cap \{y, \overline{y} \mid y \in \mathbf{Y}\} = \emptyset$, we have that whatever assignment on the variables $\mathbf{X}$ and $\mathbf{Z}$ is fixed via the atoms in D (in particular, also when $D = \emptyset$), there will always be an assignment on the variables $\mathbf{Y}$, represented by the atoms $\{y, \overline{y} \mid y \in \mathbf{Y}\}$, such that the atom *sat* will be derived in the GL reduct of $\Pi_1 \cup D$, irrespective of the guess Φ, and hence the assignment on the variables $\mathbf{X}$. Hence, again, $\Pi_1 \cup D$ and $\Pi_2 \cup D$ have no CWVs; a contradiction.

We thus have that D cannot be empty, but also cannot contain any atoms from $\mathcal{A}$, and hence, cannot exist. Since, by construction, all CWVs of Π_1 and Π_2 are also WVs (as the respective epistemic guesses are never in a subset-relationship), the above holds for both uniform CWV- and uniform WV-equivalence. This concludes the proof. □

From the proof of the above theorem, we immediately obtain the following statement for uniform WV-equivalence, which follows from the fact that our lower-bound construction employs an encoding for 3-QBF where the set of WVs and CWVs always coincide, and hence, the two ELPs Π_1 and Π_2 in this construction are uniformly WV-equivalent iff they are uniformly CWV-equivalent.

Theorem 17
Deciding uniform WV-equivalence for two ELPs is Π_3^P-hard.

Note, however, that our upper bound construction does not give an upper bound for uniform WV-equivalence, since verifying that some CWV is a WV is Π_3^P-hard itself (Shen and Eiter 2016).

5 Conclusions

In this paper, we have defined and studied the notion of uniform equivalence for epistemic logic programs. Programs are uniformly WV- or CWV-equivalent if they yield the same world views or candidate world views, respectively. In contrast to strong equivalence for ELPs, the two notions (for WV and CWV) do not coincide, but they generalize uniform equivalence for standard logic programs interpreted using the Answer Set semantics. We also provided a characterization of both notions of uniform equivalence by means of a UE-function, in the spirit of the SE-function of (Faber et al. 2019). However, unlike the SE-function this characterization is relatively straightforward and provides only little insight into the problem. While this reduces the potential impact of the characterization, it appears that one cannot do better. In fact, we show that deciding uniform equivalence on ELPs is at least Π_3^P-hard and thus probably much harder than deciding strong equivalence

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

on ELPs, which is coNP-complete. This result provides a further indication that a more compact representation of the UE-function is unlikely to exist.

For future work, it would be interesting to see whether other forms of equivalences between ELPs exist that are less restrictive than strong equivalence but more restrictive than uniform equivalence, and, ideally, for which the decision problem also lies between the respective complexities of deciding uniform and strong equivalence.

Acknowledgements

Wolfgang Faber and Michael Morak were supported by the S&T Cooperation CZ 05/2019 "Identifying Undoable Actions and Events in Automated Planning by Means of Answer Set Programming." Stefan Woltran was supported by the Austrian Science Fund (FWF) under grant number Y698.

References

Bichler, M., Morak, M., and Woltran, S. 2018. Single-shot epistemic logic program solving. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, J. Lang, Ed. ijcai.org, 1714–1720.

Brewka, G., Eiter, T., and Truszczynski, M. 2011. Answer set programming at a glance. *Commun. ACM 54,* 12, 92–103.

Cabalar, P., Pearce, D., and Valverde, A. 2007. Minimal logic programs. In *Logic Programming, 23rd International Conference, ICLP 2007, Porto, Portugal, September 8-13, 2007, Proceedings*, V. Dahl and I. Niemelä, Eds. Lecture Notes in Computer Science, vol. 4670. Springer, 104–118.

Eiter, T. and Fink, M. 2003. Uniform equivalence of logic programs under the stable model semantics. In *Proc. ICLP*. 224–238.

Eiter, T., Fink, M., Pührer, J., Tompits, H., and Woltran, S. 2013. Model-based recasting in answer-set programming. *Journal of Applied Non-Classical Logics 23,* 1-2, 75–104.

Eiter, T., Fink, M., Tompits, H., and Woltran, S. 2004. Simplifying logic programs under uniform and strong equivalence. In *Proc. LPNMR*. 87–99.

Eiter, T., Fink, M., Tompits, H., and Woltran, S. 2005. Strong and uniform equivalence in answer-set programming: Characterizations and complexity results for the non-ground case. In *Proc. AAAI*. 695–700.

Eiter, T. and Gottlob, G. 1995. On the computational cost of disjunctive logic programming: Propositional case. *Ann. Math. Artif. Intell. 15,* 3-4, 289–323.

Faber, W., Morak, M., and Woltran, S. 2019. Strong equivalence for epistemic logic programs made easy. In *Proc. AAAI*.

Fariñas del Cerro, L., Herzig, A., and Su, E. I. 2015. Epistemic equilibrium logic. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, Q. Yang and M. Wooldridge, Eds. AAAI Press, 2964–2970.

Gebser, M., Kaminski, R., Kaufmann, B., and Schaub, T. 2012. *Answer Set Solving in Practice*. Morgan & Claypool.

Gelfond, M. 1991. Strong introspection. In *Proceedings of the 9th National Conference on Artificial Intelligence, Anaheim, CA, USA, July 14-19, 1991, Volume 1*, T. L. Dean and K. R. McKeown, Eds. AAAI Press / The MIT Press, 386–391.

Gelfond, M. 1994. Logic programming and reasoning with incomplete information. *Ann. Math. Artif. Intell. 12,* 1-2, 89–116.

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press

Gelfond, M. 2011. New semantics for epistemic specifications. In *Logic Programming and Nonmonotonic Reasoning - 11th International Conference, LPNMR 2011, Vancouver, Canada, May 16-19, 2011. Proceedings*, J. P. Delgrande and W. Faber, Eds. Lecture Notes in Computer Science, vol. 6645. Springer, 260–265.

Gelfond, M. and Lifschitz, V. 1988. The stable model semantics for logic programming. In *Logic Programming, Proceedings of the Fifth International Conference and Symposium, Seattle, Washington, USA, August 15-19, 1988 (2 Volumes)*, R. A. Kowalski and K. A. Bowen, Eds. MIT Press, 1070–1080.

Gelfond, M. and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Comput. 9,* 3/4, 365–386.

Janhunen, T., Oikarinen, E., Tompits, H., and Woltran, S. 2009. Modularity aspects of disjunctive stable models. *J. Artif. Intell. Res. 35*, 813–857.

Kahl, P. T. 2014. Refining the semantics for epistemic logic programs. Ph.D. thesis, Texas Tech University, Texas, USA.

Kahl, P. T., Watson, R., Balai, E., Gelfond, M., and Zhang, Y. 2015. The language of epistemic specifications (refined) including a prototype solver. *J. Log. Comput. 25.*

Lifschitz, V., Pearce, D., and Valverde, A. 2001. Strongly equivalent logic programs. *ACM Trans. Comput. Log. 2,* 4, 526–541.

Lifschitz, V., Tang, L. R., and Turner, H. 1999. Nested expressions in logic programs. *Ann. Math. Artif. Intell. 25,* 3-4, 369–389.

Lifschitz, V. and Turner, H. 1994. Splitting a logic program. In *Proc. ICLP*. 23–37.

Lin, F. and Chen, Y. 2007. Discovering classes of strongly equivalent logic programs. *J. Artif. Intell. Res. 28*, 431–451.

Maher, M. J. 1988. Equivalences of Logic Programs. In *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed. Morgan Kaufmann Publishers, Inc., Washington DC, 627–658.

Oikarinen, E. 2007. Modular answer set programming. In *Proc. ICLP*. 462–463.

Pearce, D., Tompits, H., and Woltran, S. 2009. Characterising equilibrium logic and nested logic programs: Reductions and complexity. *TPLP 9,* 5, 565–616.

Sagiv, Y. 1988. Optimizing Datalog Programs. In *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed. Morgan Kaufmann Publishers, Inc., Washington DC, 659–698.

Schaub, T. and Woltran, S. 2018. Special issue on answer set programming. *KI 32,* 2-3.

Shen, Y. and Eiter, T. 2016. Evaluating epistemic negation in answer set programming. *Artif. Intell. 237*, 115–135.

Son, T. C., Le, T., Kahl, P. T., and Leclerc, A. P. 2017. On computing world views of epistemic logic programs. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, C. Sierra, Ed. ijcai.org, 1269–1275.

Truszczynski, M. 2011. Revisiting epistemic specifications. In *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning - Essays Dedicated to Michael Gelfond on the Occasion of His 65th Birthday.*

Turner, H. 2003. Strong equivalence made easy: nested expressions and weight constraints. *TPLP 3,* 4-5, 609–622.

Wang, K. and Zhang, Y. 2005. Nested epistemic logic programs. In *Logic Programming and Nonmonotonic Reasoning, 8th International Conference, LPNMR 2005, Diamante, Italy, September 5-8, 2005, Proceedings*, C. Baral, G. Greco, N. Leone, and G. Terracina, Eds. LNCS, vol. 3662. Springer, 279–290.

https://doi.org/10.1017/S1471068419000218 Published online by Cambridge University Press