INFSYS RESEARCH REPORT

INFSYS RESEARCH REPORT 1843-06-08, MARCH 2010

A NOVEL COMBINATION OF ANSWER SET PROGRAMMING WITH DESCRIPTION LOGICS FOR THE SEMANTIC WEB

MARCH 20, 2010

Thomas Lukasiewicz [1]

Abstract. We present a novel combination of disjunctive programs under the answer set semantics with description logics for the Semantic Web. The combination is based on a well-balanced interface between disjunctive programs and description logics, which guarantees the decidability of the resulting formalism without assuming syntactic restrictions. We show that the new formalism has very nice semantic properties. In particular, it faithfully extends both disjunctive programs and description logics. Furthermore, we describe algorithms for reasoning in the new formalism, and we give a precise picture of its computational complexity. We also define the well-founded semantics for the normal case, where normal programs are combined with tractable description logics, and we explore its semantic and computational properties. In particular, we show that the well-founded semantics approximates the answer set semantics. We also describe algorithms for the problems of consistency checking and literal entailment under the well-founded semantics, and we give a precise picture of their computational complexity. As a crucial property, in the normal case, consistency checking and literal entailment under the well-founded semantics are both tractable in the data complexity, and even first-order rewritable (and thus can be done in LOGSPACE in the data complexity) in a special case that is especially useful for representing mappings between ontologies.

[1]Computing Laboratory, University of Oxford, Wolfson Building, Parks Road, Oxford OX1 3QD, UK; e-mail: thomas.lukasiewicz@comlab.ox.ac.uk. Institut für Informationssysteme, Technische Universität Wien, Favoritenstraße 9-11, 1040 Wien, Austria; e-mail: thomas.lukasiewicz@kr.tuwien.ac.at.

Acknowledgements: This work has been supported by the German Research Foundation (DFG) under the Heisenberg Programme.

Contents

1 Introduction

The *Semantic Web* [7, 28] aims at an extension of the current World Wide Web by standards and technologies that help machines to understand the information on the Web so that they can support richer discovery, data integration, navigation, and automation of tasks. The main ideas behind it are to add a machine-readable meaning to Web pages, to use ontologies for a precise definition of shared terms in Web resources, to use knowledge representation technology for automated reasoning from Web resources, and to apply cooperative agent technology for processing the information of the Web.

The Semantic Web consists of several hierarchical layers, where the *Ontology layer*, in form of the *OWL Web Ontology Language* [63, 35, 5], is currently the highest layer of sufficient maturity. OWL consists of three increasingly expressive sublanguages, namely *OWL Lite*, *OWL DL*, and *OWL Full*. OWL Lite and OWL DL are essentially very expressive description logics with an RDF syntax [35]. As shown in [33], ontology entailment in OWL Lite (resp., OWL DL) reduces to knowledge base (un)satisfiability in the description logic $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$). As a next important step in the development of the Semantic Web, one aims at sophisticated representation and reasoning capabilities for the *Rules*, *Logic*, and *Proof layers* of the Semantic Web.

In particular, there is a large body of work on integrating rules and ontologies, which is a key requirement of the layered architecture of the Semantic Web. Significant research efforts focus on hybrid integrations of rules and ontologies, called *description logic programs* (or *dl-programs*), which are of the form $KB = (L, P)$, where L is a description logic knowledge base and P is a finite set of rules involving either queries to L in a loose integration, or concepts and roles from L as unary and binary predicates, respectively, in a tight integration (see especially [4, 22, 21, 19, 57] for recent surveys).

However, especially the tight integration of rules and ontologies presents many semantic and computational difficulties [58]. As many expressive description logics are very close to the decidability / undecidability frontier (such as $\mathcal{SHOIN}(\mathbf{D})$, which is only decidable when number restrictions are limited to simple abstract roles [36]), developing decidable extensions of them by rules turns out to be a naturally hard task, and often comes along with strong syntactic restrictions on the resulting language (such as syntactic safety conditions and/or syntactic partitionings of the vocabulary).

Nonetheless, in rule-based systems in the Semantic Web, we would like to use vocabulary from formal ontologies, and we would like to do it without syntactic restrictions. In this paper, we show that the main difficulties with the above tight integrations of rules and ontologies lies actually in the *perspective of the integration*. That is, they all look from the *perspective of description logics* at the integration of rules and ontologies. However, for extending certain kinds of rule-based systems by vocabulary from ontologies, we actually do not need the full power of a rule-based extension of description logics. This is the main idea behind this paper. More precisely, we look at the integration of rules and ontologies from the *perspective of rule-based systems*. The main contributions of this paper can be summarized as follows:

- We present a novel combination of disjunctive logic programs under the answer set semantics with description logics. In detail, we present a novel form of tightly integrated disjunctive dl-programs $KB = (L, P)$ under the answer set semantics, which allows for decidable reasoning, without assuming any syntactic restrictions (see Sections 2 and 10 for a detailed comparison to previous approaches to dl-programs). Intuitively, the main idea behind the semantics of the new dl-programs $KB = (L, P)$ is to interpret P relative to Herbrand interpretations that also satisfy L, while L is interpreted relative to general interpretations over a first-order domain. That is, we modularly combine the standard semantics of disjunctive logic programs P and of description logics L, via a well-balanced interface between P and L.

- We show that the new approach to disjunctive dl-programs under the answer set semantics has very nice semantic features. In particular, the answer set semantics faithfully extends both disjunctive logic programs under the answer set semantics and description logics under the standard first-order semantics, and its closed-world property is limited to explicit default-negated atoms in rule bodies. Furthermore, the new approach does not need the unique name assumption. We also analyze the computational aspects of the new formalism. We describe algorithms for deciding answer set existence, brave consequences, and cautious consequences. This shows in particular that these decision problems are all decidable. We also draw a precise picture of their complexity.

- We also define the well-founded semantics for the special case of normal dl-programs, and explore its semantic and computational properties. In particular, we show that the well-founded semantics faithfully extends its classical counterpart for ordinary normal logic programs, and that it approximates the answer set semantics. We also describe algorithms for consistency checking and literal entailment under the well-founded semantics, and we analyze the data and general complexity of these two central computational problems. As a crucial property, normal dl-programs under the well-founded semantics allow for tractable consistency checking and for tractable literal entailment in the data complexity, and they have even a first-order rewritable (and thus LOGSPACE data complexity) special case, which is especially interesting for representing (deterministic) ontology mappings.

The rest of this paper is organized as follows. In Section 2, we describe the key ideas behind the new formalism of this paper. Section 3 recalls disjunctive and normal programs under the answer set semantics and under the well-founded semantics, respectively, while Section 4 recalls the expressive description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$ as well as the tractable description logic *DL-Lite*$_\mathcal{A}$. In Section 5, we introduce our novel approach to disjunctive and normal dl-programs under the answer set semantics and under the well-founded semantics, respectively, and in Section 6, we analyze its semantic properties. Sections 7 to 9 focus on the computational properties, including a first-order rewritable special case. In Section 10, we discuss related work. Section 11 summarizes our main results and gives an outlook on future research. Detailed proofs of all results are given in Appendices A to D.

2 Motivating Example and Key Ideas

In this section, we illustrate the key ideas behind the tight combination of disjunctive logic programs with description logics that we elaborate and explore in this paper. We first provide an example of this combination as follows.

Example 2.1 Suppose that we use a disjunctive logic program to describe the paper assignment in a reviewing process. The following collection of rules may encode that (1) candidate reviewers for a paper are all those referees who are experts in an area of the paper and who are not in a conflict situation on this paper, (2) an expert in an area is someone who has written at least three papers in that area, (3) someone is in a conflict situation on a paper if she is a co-author of an author of the paper, (4) any two authors of the same paper are co-authors, (5) a referee is either a senior or junior scientist, (6) the paper p_0 is in the Semantic Web (SW) area, and John is a referee, who has written the three papers p_1, p_2, and p_3, which are all in the

Semantic Web area, and (7) the paper p_4 lies either in the Semantic Web or in the database (DB) area:

(1) $\mathit{cand}(X, Q) \leftarrow \mathit{paperArea}(Q, A), \mathit{referee}(X), \mathit{expert}(X, A), \mathit{not}\ \mathit{conflict}(X, Q);$
(2) $\mathit{expert}(X, A) \leftarrow \mathit{isAuthorOf}(X, Q_1), \mathit{isAuthorOf}(X, Q_2), \mathit{isAuthorOf}(X, Q_3), \mathit{inArea}(Q_1, A), \mathit{inArea}(Q_2, A), \mathit{inArea}(Q_3, A), Q_1 \neq Q_2, Q_2 \neq Q_3, Q_1 \neq Q_3;$
(3) $\mathit{conflict}(X, Q) \leftarrow \mathit{co\text{-}author}(X, Y), \mathit{isAuthorOf}(Y, Q);$
(4) $\mathit{co\text{-}author}(X, Y) \leftarrow \mathit{isAuthorOf}(X, Q), \mathit{isAuthorOf}(Y, Q);$
(5) $\mathit{senior}(X) \vee \mathit{junior}(X) \leftarrow \mathit{referee}(X);$
(6) $\mathit{referee}(\mathit{john});\ \mathit{isAuthorOf}(\mathit{john}, p_1);\ \mathit{isAuthorOf}(\mathit{john}, p_2);\ \mathit{isAuthorOf}(\mathit{john}, p_3);\ \mathit{paperArea}(p_0, \text{“}SW\text{”});\ \mathit{inArea}(p_1, \text{“}SW\text{”});\ \mathit{inArea}(p_2, \text{“}SW\text{”});\ \mathit{inArea}(p_3, \text{“}SW\text{”});$
(7) $\mathit{inArea}(p_4, \text{“}SW\text{”}) \vee \mathit{inArea}(p_4, \text{“}DB\text{”})\ .$

The predicates in the above rules, however, are naturally related via additional ontological knowledge about scientists and their publications. For example, the following description logic axioms may encode that (8) conference and journal papers are articles, (9) conference papers are not journal papers, (10) *isAuthorOf* relates scientists and articles, (11) *isAuthorOf* is the inverse of *hasAuthor*, i.e., (*scientist*, *article*) belongs to *isAuthorOf* iff (*article*, *scientist*) belongs to *hasAuthor*, (12) *hasFirstAuthor* is a functional binary relationship, and (13) the individual i_1 is a scientist whose name is *mary* and who is the author of article i_2, which is entitled "*Ontology Languages*" and has been published in the year 2008:

(8) $\mathit{ConferencePaper} \sqsubseteq \mathit{Article};\ \mathit{JournalPaper} \sqsubseteq \mathit{Article};$
(9) $\mathit{ConferencePaper} \sqsubseteq \neg \mathit{JournalPaper};$
(10) $\exists \mathit{isAuthorOf} \sqsubseteq \mathit{Scientist};\ \exists \mathit{isAuthorOf}^- \sqsubseteq \mathit{Article};$
(11) $\mathit{isAuthorOf}^- \sqsubseteq \mathit{hasAuthor};\ \mathit{hasAuthor}^- \sqsubseteq \mathit{isAuthorOf};$
(12) $(\mathsf{funct}\ \mathit{hasFirstAuthor});$
(13) $\mathit{Scientist}(i_1);\ \mathit{name}(i_1, \mathit{mary});\ \mathit{isAuthorOf}(i_1, i_2);\ \mathit{Article}(i_2);\ \mathit{title}(i_2, \text{“}\mathit{Ontology\ Languages}\text{”});\ \mathit{yearOfPublication}(i_2, 2008)\ .$

In this paper, we elaborate two different semantics for such (tight) combinations of disjunctive logic programs with description logics, namely, an answer set semantics for the general case, and a well-founded semantics for the combination of normal logic programs with certain description logics. Intuitively, description logic knowledge bases are used to further constrain the models of the disjunctive (resp., normal) logic program under the answer set (resp., well-founded) semantics. This combination will have no syntactic restrictions (such as syntactic safety conditions and/or syntactic partitionings of the vocabulary) on the disjunctive (resp., normal) logic programs, but in the same time very nice computational properties (including decidability and special-case tractability).

The above tight combination of disjunctive logic programs P with description logic knowledge bases L is much different from the loose integration introduced in [25, 21], where rule bodies in P may contain

queries to L as interfaces between P and L (allowing for a flow of information from L to P, and from P to L via query arguments), and where it is not possible to use concepts and roles from L as predicates in P, like here. The following example illustrates this (syntactic) difference and shows the advantages and flexibility of the tight integration (compared to the loose on in [25, 21]).

Example 2.2 Consider again the disjunctive program P and the description logic knowledge base L of Example 2.1. Observe that the predicate symbol *isAuthorOf* in P is also a role in L, and it freely occurs in both rule bodies and rule heads in P (which is both not possible in [25, 21]). Furthermore, we can easily use L to express additional constraints on the predicate symbols in P. For example, we may use the two concept inclusion axioms $\exists \mathit{conflict} \sqsubseteq \mathit{Scientist}$ and $\exists \mathit{conflict}^{-1} \sqsubseteq \mathit{Article}$ in L to express that the relationship for conflict situations in P relates only scientists and articles.

In addition, using queries to L in rule bodies in P in [25, 21] has also a different semantics than using concepts and roles from L as predicates in rule bodies and heads in P. This (semantic) difference is illustrated by the following example.

Example 2.3 The combination of

$$\begin{aligned} L &= \{\mathit{person}(a),\ \mathit{person} \sqsubseteq \mathit{male} \sqcup \mathit{female}\} \text{ and} \\ P &= \{\mathit{referee}(X) \leftarrow \mathit{male}(X),\ \mathit{referee}(X) \leftarrow \mathit{female}(X)\} \end{aligned}$$

here implies the ground atom $\mathit{referee}(a)$, while the one of

$$\begin{aligned} L' &= \{\mathit{person}(a),\ \mathit{person} \sqsubseteq \mathit{male} \sqcup \mathit{female}\} \text{ and} \\ P' &= \{\mathit{referee}(X) \leftarrow \mathit{DL}[\mathit{male}](X),\ \mathit{referee}(X) \leftarrow \mathit{DL}[\mathit{female}](X)\} \end{aligned}$$

as in [25, 21] does *not*, since the two queries $\mathit{DL}[\mathit{male}](X)$ and $\mathit{DL}[\mathit{female}](X)$ are evaluated independently from each other, and neither $\mathit{male}(a)$ nor $\mathit{female}(a)$ follows from L'. To obtain the conclusion $\mathit{referee}(a)$ in [25, 21], one has to directly use the rule $\mathit{referee}(X) \leftarrow \mathit{DL}[\mathit{male} \sqcup \mathit{female}](X)$.

3 Disjunctive Programs

In this section, we recall disjunctive and normal programs (with default negation) under the answer set semantics and under the well-founded semantics, respectively; see especially [41] and [62], respectively, for further details and background.

3.1 Syntax

Let Φ be a first-order vocabulary with nonempty finite sets of constant and predicate symbols, but no function symbols. Let $\mathcal{X}$ be a set of variables. A *term* is either a variable from $\mathcal{X}$ or a constant symbol from Φ. An *atom* is of the form $p(t_1, \ldots, t_n)$, where p is a predicate symbol of arity $n \geqslant 0$ from Φ, and $t_1, \ldots, t_n$ are terms. A *literal* l is an atom p or a negated atom $\mathit{not}\, p$. A *disjunctive rule* (or simply *rule*) r is of the form

$$\alpha_1 \vee \cdots \vee \alpha_k \leftarrow \beta_1, \ldots, \beta_n, \mathit{not}\, \beta_{n+1}, \ldots, \mathit{not}\, \beta_{n+m}\,, \tag{1}$$

where $\alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_{n+m}$ are atoms and $k, m, n \geqslant 0$. We call $\alpha_1 \vee \cdots \vee \alpha_k$ the *head* of r, while the conjunction $\beta_1, \ldots, \beta_n, \mathit{not}\, \beta_{n+1}, \ldots, \mathit{not}\, \beta_{n+m}$ is its *body*. We define $H(r) = \{\alpha_1, \ldots, \alpha_k\}$ and

$B(r) = B^+(r) \cup B^-(r)$, where $B^+(r) = \{\beta_1, \ldots, \beta_n\}$ and $B^-(r) = \{\beta_{n+1}, \ldots, \beta_{n+m}\}$. A rule of the form (1) with $m = n = 0$ is also called a *fact*. A *disjunctive program* P is a finite set of disjunctive rules of the form (1). We say P is *positive* iff $m = 0$ for all disjunctive rules (1) in P. We say P is a *normal program* iff $k \leqslant 1$ for all disjunctive rules (1) in P.

Example 3.1 An online store (such as *amazon.com*) may use the subsequent normal program P to express that (1) pc_1 and pc_2 are personal computers, (2) pc_1 and obj_3 are brand new, (3) *dell* is the vendor of pc_1 and pc_2, (4) a customer avoids all cameras not on offer, (5) all electronic products that are not brand new are on offer, (6) each vendor of a product is a provider, (7) each entity providing a product is a provider, (8) all related products are similar, and (9) the binary similarity relation on products is transitively closed:

(1) $pc(pc_1)$; $pc(pc_2)$;
(2) $brand\_new(pc_1)$; $brand\_new(obj_3)$;
(3) $vendor(dell, pc_1)$; $vendor(dell, pc_2)$;
(4) $avoid(X) \leftarrow camera(X), not\ offer(X)$;
(5) $offer(X) \leftarrow electronics(X), not\ brand\_new(X)$;
(6) $provider(V) \leftarrow vendor(V, X), product(X)$;
(7) $provider(V) \leftarrow provides(V, X), product(X)$;
(8) $similar(X, Y) \leftarrow related(X, Y)$;
(9) $similar(X, Z) \leftarrow similar(X, Y), similar(Y, Z)$.

In a disjunctive program P', we may additionally express that (10) obj_3 is either a personal computer or a laptop, and that (11) every screen is either a TFT, a CRT, or a touchscreen:

(10) $pc(obj_3) \vee laptop(obj_3)$;
(11) $tft(X) \vee crt(X) \vee touchscreen(X) \leftarrow screen(X)$.

3.2 Answer Set Semantics

The answer set semantics of disjunctive programs is defined in terms of finite sets of ground atoms, which represent Herbrand interpretations. Positive disjunctive programs are associated with all their minimal satisfying sets of ground atoms, while the semantics of general disjunctive programs is defined by reduction to the minimal model semantics of positive disjunctive programs via the Gelfond-Lifschitz reduct [29].

The *Herbrand universe* of a disjunctive program P, denoted HU_P, is the set of all constant symbols appearing in P. If there is no such constant symbol, then $HU_P = \{c\}$, where c is an arbitrary constant symbol from Φ. As usual, terms, atoms, literals, rules, programs, etc. are *ground* iff they do not contain any variables. The *Herbrand base* of a disjunctive program P, denoted HB_P, is the set of all ground atoms that can be constructed from the predicate symbols appearing in P and the constant symbols in HU_P. Hence, in the standard answer set semantics, the Herbrand base is constructed from all constant and predicate symbols in a given disjunctive program, and thus the Herbrand base is finite. A *ground instance* of a rule $r \in P$ is obtained from r by replacing every variable that occurs in r by a constant symbol from HU_P. We denote by $ground(P)$ the set of all ground instances of rules in P.

An *interpretation* I relative to a disjunctive program P is a subset of HB_P. Informally, every such I represents the Herbrand interpretation in which all $a \in I$ (resp., $a \in HB_P - I$) are true (resp., false). An interpretation I is a *model* of a ground atom $a \in HB_P$, or I *satisfies* a, denoted $I \models a$, iff $a \in I$. We say

I is a *model* of a ground rule r, denoted $I \models r$, iff $I \models \alpha$ for some $\alpha \in H(r)$ whenever $I \models B(r)$, that is, $I \models \beta$ for all $\beta \in B^+(r)$ and $I \not\models \beta$ for all $\beta \in B^-(r)$. We say I is a *model* of a disjunctive program P, denoted $I \models P$, iff $I \models r$ for every $r \in ground(P)$.

An *answer set* of a positive disjunctive program P is a minimal model of P relative to set inclusion. The *Gelfond-Lifschitz reduct* of a disjunctive program P relative to $I \subseteq HB_P$, denoted P^I, is the ground positive disjunctive program obtained from $ground(P)$ by

(i) deleting every rule r such that $B^-(r) \cap I \neq \emptyset$, and

(ii) deleting the negative body from each remaining rule.

An *answer set* of a disjunctive program P is an interpretation $I \subseteq HB_P$ such that I is an answer set of P^I. A disjunctive program P is *consistent* iff P has an answer set. Hence, under the answer set semantics, every disjunctive program P is interpreted as its grounding $ground(P)$. Note that the answer sets of any disjunctive program P are also minimal models of P. An equivalent definition of the answer set semantics is based on the so-called *FLP-reduct* [27]: The *FLP-reduct* of a disjunctive program P relative to $I \subseteq HB_P$, denoted P^I, is the set of all $r \in ground(P)$ such that $I \models B(r)$. An interpretation $I \subseteq HB_P$ is an *answer set* of P iff I is a minimal model of P^I.

We finally recall the notions of *cautious* (resp., *brave*) *reasoning* from disjunctive programs under the answer set semantics. A ground atom $a \in HB_P$ is a *cautious* (resp., *brave*) *consequence* of a disjunctive program P under the answer set semantics iff every (resp., some) answer set of P satisfies a.

Example 3.2 *Let the disjunctive program P'' be given by the disjunctive program P' of Example 3.1 and the facts camera(cam), electronics(cam), and brand_new(cam). Then, P'' has two different answer sets. They contain the facts in lines (1) to (3) of Example 3.1, the above ones, avoid(cam), and either pc(obj_3) or laptop(obj_3). Hence, all the former but the last two facts are cautious consequences of P'', while pc(obj_3) and laptop(obj_3) are brave consequences of P''.*

Observe that for positive disjunctive programs P, since the set of all answer sets of P is given by the set of all minimal models of P, it holds that $a \in HB_P$ is a cautious consequence of P under the answer set semantics iff a is a logical consequence of the propositional positive disjunctive program $ground(P)$. Note that, more generally, this result holds also when a is a ground formula constructed from HB_Φ using the Boolean operators $\wedge$ and $\vee$. That is, the *closed-world property* (that is, the derivation of negative facts from the absence of derivations of positive facts) of the above notion of cautious reasoning under the answer set semantics is actually limited to the occurrences of default negations in rule bodies.

3.3 Well-Founded Semantics

Besides the answer set semantics, the *well-founded semantics* [62] is the most widely used semantics for nonmonotonic logic programs, and it is especially under a data-oriented perspective of great importance for the Web. As nice features, the well-founded semantics is defined for all normal programs (unlike the answer set semantics), has a polynomial data tractability (while the answer set semantics is intractable), approximates the answer set semantics (in the sense that the well-founded semantics is a subset of every answer set), and coincides with the canonical model of stratified programs. The well-founded semantics of normal programs P has many different equivalent definitions [62, 6]. We recall here the one based on unfounded sets, via the operators U_P, T_P, and W_P.

We first give some preliminary definitions. For literals $l = a$ (resp., $l = \neg a$), we use $\neg.l$ to denote $\neg a$ (resp., a), and for sets of literals S, we define $\neg.S = \{\neg.l \mid l \in S\}$ and $S^+ = \{a \in S \mid a \text{ is an atom}\}$. We

denote by $Lit_P = HB_P \cup \neg.HB_P$ the set of all ground literals with predicate and constant symbols from P. A set of ground literals $S \subseteq Lit_P$ is *consistent* iff $S \cap \neg.S = \emptyset$. A *(three-valued) interpretation* relative to P is any consistent set of ground literals $I \subseteq Lit_P$.

We next define the notion of an unfounded set. A set $U \subseteq HB_P$ is an *unfounded set* of P relative to $I \subseteq Lit_P$ iff for every $a \in U$ and every $r \in ground(P)$ with $H(r) = a$, either

(i) $\neg b \in I \cup \neg.U$ for some atom $b \in B^+(r)$, or

(ii) $b \in I$ for some atom $b \in B^-(r)$.

There exists the greatest unfounded set of P relative to I, denoted $U_P(I)$. Intuitively, if I is compatible with P, then all atoms in $U_P(I)$ can be safely switched to false and the resulting interpretation is still compatible with P. The greatest unfounded set of a partial interpretation I intuitively collects all those atoms that cannot become true when extending I with further information. An atom b is unfounded iff there is no rule with b in its head and with a body that can be made true. For example, an atom not appearing in any head is clearly unfounded. Observe that the falsity of rule bodies can be testified by unfounded atoms belonging to the same unfounded set, giving a notion of "self-supportedness".

We are now ready to define the two operators T_P and W_P on consistent $I \subseteq Lit_P$ as follows:

- $T_P(I) = \{H(r) \mid r \in ground(P),\ B^+(r) \cup \neg.B^-(r) \subseteq I\}$;
- $W_P(I) = T_P(I) \cup \neg.U_P(I)$.

The operator W_P is monotonic, and thus has a least fixpoint, denoted $lfp(W_P)$, which is the *well-founded semantics* of P, denoted $WFS(P)$. A ground atom $a \in HB_P$ is *well-founded* (resp., *unfounded*) relative to P, if a (resp., $\neg a$) is in $lfp(W_P)$. Intuitively, starting with $I = \emptyset$, rules are applied to obtain new positive and negated facts (via $T_P(I)$ and $\neg.U_P(I)$, respectively). This process is repeated until no longer possible. A literal $\ell \in HB_P \cup \neg.HB_P$ is a *consequence* of a normal program P under the well-founded semantics iff $\ell \in WFS(P)$.

Example 3.3 *Let the normal program P''' be given by the normal program P of Example 3.1 and the facts camera(cam), electronics(cam), and brand_new(cam). Then, $WFS(P''')$ contains all the facts in lines (1) to (3) of Example 3.1, the above ones, avoid(cam), and the negations $\neg a$ of all other atoms $a \in HB_P$. Hence, all the above literals are consequences of P''' under the well-founded semantics.*

4 Description Logics

In this section, we recall the expressive description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$, which stand behind the web ontology languages OWL Lite and OWL DL [33], respectively. Furthermore, we recall the tractable description logic *DL-Lite*$_\mathcal{A}$ [55], which adds datatypes to a restricted combination of the tractable description logics *DL-Lite*$_\mathcal{F}$ and *DL-Lite*$_\mathcal{R}$. All these description logics belong to the *DL-Lite* family [14], which are a class of restricted description logics for which the main reasoning tasks are possible in polynomial time in general and some of them even in LOGSPACE in the data complexity. The *DL-Lite* description logics are fragments of OWL and the most common tractable ontology languages in the Semantic Web context. They are especially directed towards data-intensive applications.

Intuitively, description logics model a domain of interest in terms of concepts and roles, which represent classes of individuals and binary relations between classes of individuals, respectively. A description logic knowledge base encodes especially subset relationships between concepts, subset relationships between roles, the membership of individuals to concepts, and the membership of pairs of individuals to roles.

4.1 DLs Behind OWL Lite and OWL DL

We first recall the expressive description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$ underlying the web ontology languages OWL Lite and OWL DL [33], respectively.

Syntax. We first describe the syntax of $\mathcal{SHOIN}(\mathbf{D})$. We assume a set of *elementary datatypes* and a set of *data values* $\mathbf{V}$. A *datatype* is either an elementary datatype or a set of data values (called *datatype oneOf*). A *datatype theory* $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a *datatype domain* $\Delta^{\mathbf{D}}$ and a mapping $\cdot^{\mathbf{D}}$ that assigns to each elementary datatype a subset of $\Delta^{\mathbf{D}}$ and to each data value an element of $\Delta^{\mathbf{D}}$. The mapping $\cdot^{\mathbf{D}}$ is extended to all datatypes by $\{v_1, \ldots\}^{\mathbf{D}} = \{v_1^{\mathbf{D}}, \ldots\}$. Let $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, and $\mathbf{I}$ be pairwise disjoint (nonempty) denumerable sets of *atomic concepts*, *abstract roles*, *datatype roles*, and *individuals*, respectively. We denote by $\mathbf{R}_A^-$ the set of inverses R^- of all $R \in \mathbf{R}_A$.

A *role* is an element of $\mathbf{R}_A \cup \mathbf{R}_A^- \cup \mathbf{R}_D$. *Concepts* are inductively defined as follows. Every $\phi \in \mathbf{A}$ is a concept, and if $o_1, \ldots, o_n \in \mathbf{I}$, then $\{o_1, \ldots, o_n\}$ is a concept (called *oneOf*). If ϕ, ϕ_1, and ϕ_2 are concepts and if $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$, then also $(\phi_1 \sqcap \phi_2)$, $(\phi_1 \sqcup \phi_2)$, and $\neg\phi$ are concepts (called *conjunction*, *disjunction*, and *negation*, respectively), as well as $\exists R.\phi$, $\forall R.\phi$, $\geqslant nR$, and $\leqslant nR$ (called *exists*, *value*, *atleast*, and *atmost restriction*, respectively) for an integer $n \geqslant 0$. If D is a datatype and $U \in \mathbf{R}_D$, then $\exists U.D$, $\forall U.D$, $\geqslant nU$, and $\leqslant nU$ are concepts (called *datatype exists*, *value*, *atleast*, and *atmost restriction*, respectively) for an integer $n \geqslant 0$. We write $\top$ and $\bot$ to abbreviate the concepts $\phi \sqcup \neg\phi$ and $\phi \sqcap \neg\phi$, respectively, and we eliminate parentheses as usual.

An *axiom* has one of the following forms:

(1) $\phi \sqsubseteq \psi$ (called *concept inclusion axiom*), where ϕ and ψ are concepts;

(2) $R \sqsubseteq S$ (called *role inclusion axiom*), where either $R, S \in \mathbf{R}_A$ or $R, S \in \mathbf{R}_D$;

(3) $\mathrm{Trans}(R)$ (called *transitivity axiom*), where $R \in \mathbf{R}_A$;

(4) $\phi(a)$ (called *concept membership axiom*), where ϕ is a concept and $a \in \mathbf{I}$;

(5) $R(a, b)$ (resp., $U(a, v)$) (called *role membership axiom*), where $R \in \mathbf{R}_A$ (resp., $U \in \mathbf{R}_D$) and $a, b \in \mathbf{I}$ (resp., $a \in \mathbf{I}$ and v is a data value); and

(6) $a = b$ (resp., $a \neq b$) (*equality* (resp., *inequality*) *axiom*), where $a, b \in \mathbf{I}$.

A *(description logic) knowledge base* L is a finite set of axioms. For decidability, number restrictions in L are restricted to simple abstract roles [36].

The syntax of $\mathcal{SHIF}(\mathbf{D})$ is as the above syntax of $\mathcal{SHOIN}(\mathbf{D})$, but without the oneOf constructor and with the atleast and atmost constructors limited to 0 and 1.

Example 4.1 The subsequent description logic knowledge base L expresses that (1) textbooks are books, (2) personal computers and laptops are mutually exclusive electronic products, (3) books and electronic products are mutually exclusive products, (4) objects on offer are products, (5) every product has at least one related product, (6) only products are related to each other, (7) the relatedness between products is symmetric, (8) *tb_ai* and *tb_lp* are textbooks, (9) which are related to each other, (10) *pc_ibm* and *pc_hp* are personal computers, (11) which are related to each other, and (12) *ibm* and *hp* are providers for *pc_ibm* and *pc_hp*, respectively.

(1) *textbook* $\sqsubseteq$ *book*;
(2) *pc* $\sqcup$ *laptop* $\sqsubseteq$ *electronics*; *pc* $\sqsubseteq \neg$*laptop*;
(3) *book* $\sqcup$ *electronics* $\sqsubseteq$ *product*; *book* $\sqsubseteq \neg$*electronics*;
(4) *offer* $\sqsubseteq$ *product*;
(5) *product* $\sqsubseteq \geqslant 1$ *related*;
(6) $\geqslant 1$ *related* $\sqcup \geqslant 1$ *related*$^-$ $\sqsubseteq$ *product*;
(7) *related* $\sqsubseteq$ *related*$^-$; *related*$^-$ $\sqsubseteq$ *related*;
(8) *textbook*(*tb_ai*); *textbook*(*tb_lp*);
(9) *related*(*tb_ai*, *tb_lp*);
(10) *pc*(*pc_ibm*); *pc*(*pc_hp*);
(11) *related*(*pc_ibm*, *pc_hp*);
(12) *provides*(*ibm*, *pc_ibm*); *provides*(*hp*, *pc_hp*).

Semantics. The semantics of the description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$ is defined in terms of standard first-order interpretations as usual. An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ relative to a datatype theory $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a nonempty (*abstract*) *domain* $\Delta^{\mathcal{I}}$ disjoint from $\Delta^{\mathbf{D}}$, and a mapping $\cdot^{\mathcal{I}}$ that assigns to each atomic concept $\phi \in \mathbf{A}$ a subset of $\Delta^{\mathcal{I}}$, to each individual $o \in \mathbf{I}$ an element of $\Delta^{\mathcal{I}}$, to each abstract role $R \in \mathbf{R}_A$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and to each datatype role $U \in \mathbf{R}_D$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathbf{D}}$. We extend $\cdot^{\mathcal{I}}$ to all concepts and roles, and we define the *satisfaction* of an axiom F in an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, denoted $\mathcal{I} \models F$, as usual [33]. We say $\mathcal{I}$ *satisfies* the axiom F, or $\mathcal{I}$ is a *model* of F, iff $\mathcal{I} \models F$. We say $\mathcal{I}$ *satisfies* a knowledge base L, or $\mathcal{I}$ is a *model* of L, denoted $\mathcal{I} \models L$, iff $\mathcal{I} \models F$ for all $F \in L$. We say L is *satisfiable* (resp., *unsatisfiable*) iff L has a (resp., no) model. An axiom F is a *logical consequence* of L, denoted $L \models F$, iff each model of L satisfies F.

Example 4.2 *It is not difficult to verify that the description logic knowledge base L in Example 4.1 is satisfiable. Furthermore, it is not difficult to see that the concept inclusion axiom textbook $\sqsubseteq$ product and the concept membership axiom electronics(pc_ibm) are two logical consequences of L.*

4.2 A Tractable DL

We next recall the tractable description logic *DL-Lite*$_{\mathcal{A}}$.

Syntax. As for the elementary ingredients of *DL-Lite*$_{\mathcal{A}}$, let $\mathbf{D}$ be a finite set of *atomic datatypes* d, which are associated with pairwise disjoint sets of *data values* $\mathbf{V}_d$. Let $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, and $\mathbf{I}$ be pairwise disjoint sets of *atomic concepts*, *atomic roles*, *atomic attributes*, and *individuals*, respectively, and let $\mathbf{V}$ denote the union of all $\mathbf{V}_d$ with $d \in \mathbf{D}$.

Roles, concepts, attributes, and datatypes are as follows:

- A *basic role* Q is either an atomic role $P \in \mathbf{R}_A$ or its inverse P^-. A *(general) role* R is either a basic role Q or the negation of a basic role $\neg Q$.
- A *basic concept* B is either an atomic concept $A \in \mathbf{A}$, or an existential restriction on a basic role Q, denoted $\exists Q$, or the domain of an atomic attribute U, denoted $\delta(U)$. A *(general) concept* C is either the *universal concept* $\top_C$, or a basic concept B, or the negation of a basic concept $\neg B$, or an existential restriction on a basic role Q of the form $\exists Q.C$, where C is a concept.

- A *(general) attribute* V is either an atomic attribute U or the negation of an atomic attribute $\neg U$.
- A *basic datatype* E is the range of an atomic attribute U, denoted $\rho(U)$. A *(general) datatype* F is either the *universal datatype* $\top_D$ or an atomic datatype.

An *axiom* is of one of the following forms:

(1) $B \sqsubseteq C$ (*concept inclusion axiom*), where B is a basic concept, and C is a concept;

(2) $Q \sqsubseteq R$ (*role inclusion axiom*), where Q is a basic role, and R is a role;

(3) $U \sqsubseteq V$ (*attribute inclusion axiom*), where U is an atomic attribute, and V is an attribute;

(4) $E \sqsubseteq F$ (*datatype inclusion axiom*), where E is a basic datatype, and F is a datatype;

(5) $(\mathsf{funct}\ Q)$ (*role functionality axiom*), where Q is a basic role;

(6) $(\mathsf{funct}\ U)$ (*attribute functionality axiom*), where U is an atomic attribute;

(7) $A(a)$ (*concept membership axiom*), where A is an atomic concept and $a \in \mathbf{I}$,

(8) $P(a, b)$ (*role membership axiom*), where P is an atomic role and $a, b \in \mathbf{I}$; and

(9) $U(a, v)$ (*attribute membership axiom*), where U is an atomic attribute, $a \in \mathbf{I}$, and $v \in \mathbf{V}$.

A *TBox* is a finite set $\mathcal{T}$ of inclusion and functionality axioms such that every identifying property in $\mathcal{T}$ is primitive (see [55] for a definition of primitive identifying properties). An *ABox* $\mathcal{A}$ is a finite set of membership axioms. A *(description logic) knowledge base* $KB = \mathcal{T} \cup \mathcal{A}$ is the union of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$.

Example 4.3 *The description logic knowledge base L in Example 4.1 actually turns out to be equivalent to a description logic knowledge base L' in* DL-Lite$_\mathcal{A}$*, obtained from L by (i) breaking up all concept inclusion axioms with disjunctions into two concept inclusion axioms and (ii) replacing all number restrictions by existential restrictions.*

Semantics. The semantics of *DL-Lite*$_\mathcal{A}$ is defined in a similar way as the semantics of $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$, except that different values and different individuals also have different interpretations (*unique name assumption*).

As shown in [55], in particular, deciding the satisfiability of knowledge bases in *DL-Lite*$_\mathcal{A}$ and deciding logical consequences of membership axioms from knowledge bases in *DL-Lite*$_\mathcal{A}$ can both be done in polynomial time in general and in LOGSPACE in the size of the ABox in the data complexity.

5 Disjunctive DL-Programs

In this section, we present a novel integration of disjunctive and normal programs under the answer set semantics and under the well-founded semantics, respectively, with description logics. The basic idea behind this integration is briefly described as follows. Suppose that we have a disjunctive program P. Under the answer set semantics, P is equivalent to its grounding $ground(P)$. Suppose now that some of the ground atoms in $ground(P)$ are additionally related to each other by a description logic knowledge base L. That is, some of the ground atoms in $ground(P)$ actually represent concept and role memberships relative to L. Thus, when processing $ground(P)$, we also have to consider L. However, we only want to do it to the extent

that we actually need it for processing $ground(P)$. Hence, when taking a Herbrand interpretation $I \subseteq HB_\Phi$, we have to ensure that the ground atoms of I represent a valid constellation relative to L.

In other words, the main idea behind the semantics is to interpret P relative to Herbrand interpretations that also satisfy L, while L is interpreted relative to general interpretations over a first-order domain. Thus, we modularly combine the standard semantics of logic programs and of description logics as in [25, 21], which allows for building on the standard techniques and the results of both areas. But our new approach here allows for a much tighter integration of L and P.

5.1 Syntax

We assume a function-free first-order vocabulary Φ with nonempty finite sets of constant and predicate symbols, as in Section 3. We use Φ_c to denote the set of all constant symbols in Φ. We also assume pairwise disjoint (nonempty) denumerable sets $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, $\mathbf{I}$, and $\mathbf{V}$ of atomic concepts, abstract roles, datatype roles, individuals, and values, respectively, as in Section 4. We assume that Φ_c is a subset of $\mathbf{I} \cup \mathbf{V}$. This assumption guarantees that every ground atom constructed from atomic concepts, abstract roles, datatype roles, and constants in Φ_c can be interpreted in the description logic component. We do not assume any other restriction on the vocabularies, that is, Φ and $\mathbf{A}$ (resp., $\mathbf{R}_A \cup \mathbf{R}_D$) may have unary (resp., binary) predicate symbols in common.

A *disjunctive description logic program* (or *disjunctive dl-program*) $KB = (L, P)$ consists of a description logic knowledge base L and a disjunctive program P. It is *positive* iff P is positive. It is a *normal dl-program* iff P is a normal program.

Example 5.1 A disjunctive (resp., normal) dl-program $KB_1 = (L_1, P_1)$ (resp., $KB_2 = (L_2, P_2)$) is given by the description logic knowledge base L of Example 4.1 and the disjunctive (resp., normal) program P' (resp., P) of Example 3.1.

Another disjunctive dl-program $KB_3 = (L_3, P_3)$ is obtained from $KB_1 = (L_1, P_1)$ by adding to L_1 the concept inclusion axiom $\geqslant 1\, \mathit{similar} \sqcup \geqslant 1\, \mathit{similar}^- \sqsubseteq \mathit{product}$, which expresses that only products are similar. That is, we can easily use the description logic knowledge base L to express additional constraints on the predicate symbols in P.

The above dl-programs also show the advantages and flexibility of the tight integration between rules and ontologies (compared to the loose integration in [25, 21]): Observe that the predicate symbol *similar* in P_3 is also a role in L_3, and it freely occurs in both rule bodies and rule heads in P_3 (which is not possible in [25, 21]).

5.2 Answer Set Semantics

We now define the answer set semantics of disjunctive dl-programs via a generalization of the FLP-reduct of disjunctive programs (see Section 3).

In the sequel, let $KB = (L, P)$ be a disjunctive dl-program. A *ground instance* of a rule $r \in P$ is obtained from r by replacing every variable that occurs in r by a constant symbol from Φ_c. We denote by $ground(P)$ the set of all ground instances of rules in P. The *Herbrand base* relative to Φ, denoted HB_Φ, is the set of all ground atoms constructed with constant and predicate symbols from Φ. Observe that we now define the Herbrand base relative to Φ and not relative to P. This allows for reasoning about ground atoms from the description logic component that do not necessarily occur in P. Observe, however, that the extension from P to Φ is only a notational simplification, since we can always make constant and predicate symbols from Φ occur in P by "dummy" rules such as $\mathit{constant}(c) \leftarrow$ and $p(\boldsymbol{c}) \leftarrow p(\boldsymbol{c})$, respectively. We denote by DL_Φ

the set of all ground atoms in HB_Φ that are constructed from atomic concepts in $\mathbf{A}$, abstract roles in $\mathbf{R}_A$, concrete roles in $\mathbf{R}_D$, and constant symbols in Φ_c.

An *interpretation* I is any subset of HB_Φ. We say I is a *model* of a description logic knowledge base L, denoted $I \models L$, iff $L \cup I \cup \{\neg a \mid a \in HB_\Phi - I\}$ is satisfiable. Note that the former defines the truth of description logic knowledge bases L in Herbrand interpretations $I \subseteq HB_\Phi$ rather than first-order interpretations $\mathcal{I}$. The following lemma shows that negative concept (resp., role) membership axioms $\neg C(a)$ (resp., $\neg R(b,c)$) can be reduced to positive concept membership axioms and concept inclusion axioms.

Lemma 5.1 *Let L be a description logic knowledge base, let $C(a)$ be a concept membership axiom, and let $R(b,c)$ be a role membership axiom. Then, (a) $L \cup \{\neg C(a)\}$ is satisfiable iff $L \cup \{B(a), B \sqsubseteq \neg C\}$ is satisfiable, where B is a fresh atomic concept, and (b) $L \cup \{\neg R(b,c)\}$ is satisfiable iff $L \cup \{B(b), C(c), \exists R.C \sqsubseteq \neg B\}$ is satisfiable, where B and C are two fresh atomic concepts.*

An interpretation $I \subseteq HB_\Phi$ is a *model* of a disjunctive dl-program $KB = (L, P)$, denoted $I \models KB$, iff $I \models L$ and $I \models P$. We say KB is *satisfiable* iff it has a model.

Given a disjunctive dl-program $KB = (L, P)$, the *FLP-reduct* of KB relative to an interpretation $I \subseteq HB_\Phi$, denoted KB^I, is the disjunctive dl-program (L, P^I), where P^I is the set of all $r \in ground(P)$ with $I \models B(r)$. We define the answer set semantics of disjunctive dl-programs as follows.

Definition 5.1 (answer set semantics) *An interpretation $I \subseteq HB_\Phi$ is an* answer set *of a disjunctive dl-program KB iff I is a minimal model of KB^I. We say KB is* consistent *(resp.,* inconsistent*) iff it has an (resp., no) answer set.*

Example 5.2 *Consider again $KB_1 = (L_1, P_1)$ of Example 5.1. Then, there are two answer sets, which both contain in particular all facts in P_1 and all membership axioms in L_1, the concept membership axioms electronics(pc_ibm) and product(pc_ibm), the fact electronics(obj_3), and one contains pc(obj_3), while the other one contains laptop(obj_3).*

Cautious (resp., brave) reasoning from disjunctive dl-programs under the answer set semantics is defined as follows.

Definition 5.2 (cautious / brave consequence) *A ground atom $a \in HB_\Phi$ is a* cautious *(resp.,* brave*)* consequence *of a disjunctive dl-program KB under the answer set semantics iff every (resp., some) answer set of KB satisfies a.*

Example 5.3 *Consider again $KB_1 = (L_1, P_1)$ of Example 5.1. Then, all facts in P_1 and all membership axioms in L_1 are cautious consequences of KB_1. The concept membership axioms electronics(pc_ibm) and product(pc_ibm) as well as the fact electronics(obj_3) are other cautious consequences of KB_1, while pc(obj_3) is a brave consequence of KB_1.*

5.3 Well-Founded Semantics

We next define the well-founded semantics of normal dl-programs, which generalizes the well-founded semantics of ordinary normal programs via unfounded sets (see Section 3.3). Intuitively, the main ideas are as follows. Compared to ordinary normal programs, normal dl-programs additionally have a description logic knowledge base, which may contain disjunctive knowledge and also result into inconsistencies. We disallow such disjunctive knowledge by restricting the underlying description logic, and we handle such

inconsistencies by considering the inconsistency-generating axioms only in a final step of the definition of the well-founded semantics.

We adopt the notions of *tuple-generating dependencies*, *non-conflicting keys*, and *negative constraints* from the ontology language Datalog$^{\pm}$ [8, 9]. We assume an underlying description logic where each knowledge base L is decomposable into two disjoint subsets L^+ and L^- such that

(i) $L = L^+ \cup L^-$,

(ii) L^+ can be encoded as a set of tuple-generating dependencies, and

(iii) L^- can be encoded as a set of non-conflicting keys and negative constraints.

Note that all description logics of the *DL-Lite* family [14] have this property [8, 9], which also implies that the underlying description logic is *CWA-satisfiable* (that is, for every knowledge base L, the union of L and all negations of concept and role membership axioms not entailed by L is satisfiable). For example, in *DL-Lite*$_{\mathcal{A}}$, concept, role, and attribute inclusion axioms of the form $B \sqsubseteq \neg B'$, $Q \sqsubseteq \neg Q'$, and $U \sqsubseteq \neg U'$, respectively, can be encoded as negative constraints, while role and attribute functionality axioms $(\mathsf{funct}\ Q)$ and $(\mathsf{funct}\ U)$, respectively, can be encoded as non-conflicting keys, and all the other axioms as tuple-generating dependencies.

We use $Lit_\Phi = HB_\Phi \cup \neg.HB_\Phi$ to denote the set of all ground literals with predicate and constant symbols from Φ. A set of ground literals $S \subseteq Lit_\Phi$ is *consistent* iff $S \cap \neg.S = \emptyset$. A *(three-valued) interpretation* relative to Φ is any consistent set of ground literals $I \subseteq Lit_\Phi$. We next define the notion of an unfounded set for normal dl-programs as follows.

Definition 5.3 (unfounded set) *Let $KB = (L, P)$ be a normal dl-program, and let $I \subseteq Lit_\Phi$ be consistent. A set $U \subseteq HB_\Phi$ is an* unfounded set *of KB relative to I iff*

$(*)$ *for every $a \in U$,*

(a) for every $r \in ground(P)$ with $H(r) = a$, either

(i) $\neg b \in I \cup \neg.U$ for some $b \in B^+(r)$, or

(ii) $b \in I$ for some $b \in B^-(r)$; and

(b) $L^+ \cup S^+ \not\models a$ for every consistent $S \subseteq Lit_\Phi$ with $I \cup \neg.U \subseteq S$.

Intuitively, all the atoms of the unfounded set U of KB relative to I can be safely set to false under I. Here, compared to unfounded sets of ordinary normal programs, the condition (b) is new, which intuitively says that a will never become true via the description logic knowledge base L^+, if we expand I (to S) in a way such that all unfounded atoms are kept false. In $L^+ \cup S^+$, we only have to consider S^+, since the negated atoms in S (as long as consistent with $L^+ \cup S^+$) do not enlarge the set of positive atoms logically entailed by $L^+ \cup S^+$.

Example 5.4 Let the normal dl-program $KB = (L, P)$ be given by $L = \{q \sqsubseteq c\}$ and the following rules in P:

$$p(a) \leftarrow c(a);\ \ q(a) \leftarrow p(a);\ \ r(a) \leftarrow not\ q(a),\ not\ s(a)\,.$$

Then, $S_1 = \{p(a), q(a), c(a)\}$ is an unfounded set of KB relative to $I = \emptyset$, since $p(a)$ and $q(a)$ are unfounded due to (a.i) and their lack in $L^+ \cup S^+$ in (b), while $c(a)$ is unfounded, since no rule as in (a) defines $c(a)$, and since the sets $L^+ \cup S^+$ in (b) also do not entail $c(a)$. The set $S_2 = \{s(a)\}$ is trivially an unfounded set of KB relative to I, since neither P nor L defines $s(a)$. However, $S_3 = \{c(a)\}$ is not an unfounded set of KB relative to I, since the condition (b) fails for $c(a)$.

In the ordinary case, the set of unfounded sets of a normal program relative to I is closed under union. The following lemma shows that the same holds for normal dl-programs. That is, the set of unfounded sets of a normal dl-program relative to I is closed under union, which implies that every normal dl-program has a greatest unfounded set relative to I.

Lemma 5.2 *Let $KB = (L, P)$ be a normal dl-program, and let $I \subseteq Lit_\Phi$ be consistent. Then, the set of unfounded sets of KB relative to I is closed under union.*

Based on this result, we are now ready to generalize the operators T_P, U_P, and W_P from ordinary normal programs P to normal dl-programs KB as follows.

Definition 5.4 (T_{KB}, U_{KB}, and W_{KB}) *Let $KB = (L, P)$ be a normal dl-program. We define the operators T_{KB}, U_{KB}, and W_{KB} on all consistent $I \subseteq Lit_\Phi$ as follows:*

- $a \in T_{KB}(I)$ *iff either*
  - *(a)* $a \in HB_\Phi$ *and some* $r \in ground(P)$ *exists such that*
    - *(i)* $H(r) = a$,
    - *(ii)* $b \in I$ *for all atoms* $b \in B^+(r)$, *and*
    - *(iii)* $\neg b \in I$ *for all atoms* $b \in B^-(r)$, *or*
  - *(b)* $L^+ \cup I^+ \models a$;
- $U_{KB}(I)$ *is the greatest unfounded set of KB relative to I;*
- $W_{KB}(I) = T_{KB}(I) \cup \neg.U_{KB}(I)$.

Here, compared to the well-founded semantics of ordinary normal programs, the condition (b) $L^+ \cup I^+ \models a$ in the definition of $T_{KB}(I)$ is new. Intuitively, in addition to being implied by P under I, positive ground atoms may also be implied by L^+ under I. Note that in (b), as above, we only have to consider I^+, since the negated atoms in I (if consistent with $L^+ \cup I^+$) do not enlarge the set of positive atoms that are logically entailed by $L^+ \cup I^+$. In the ordinary case, the three operators are all monotonic. The following lemma shows that this result carries over to normal dl-programs.

Lemma 5.3 *Let KB be a normal dl-program. Then, T_{KB}, U_{KB}, and W_{KB} are all monotonic.*

Thus, in particular, W_{KB} has a least fixpoint, denoted $lfp(W_{KB})$. The well-founded semantics of normal dl-programs can thus be defined as follows.

Definition 5.5 *Let $KB = (L, P)$ be a normal dl-program. The* well-founded semantics *of KB, denoted $WFS(KB)$, is defined as $lfp(W_{KB})$, if $L \cup lfp(W_{KB})$ is satisfiable, and it is undefined, otherwise. We then say that KB is* consistent *and* inconsistent under the well-founded semantics *(or* w-consistent *and* w-inconsistent*), respectively. An atom $a \in HB_\Phi$ is* well-founded *(resp.,* unfounded*) relative to KB iff a (resp., $\neg a$) belongs to $WFS(KB)$. A literal $\ell \in HB_\Phi \cup \neg.HB_\Phi$ is a* consequence *of a normal dl-program KB under the well-founded semantics iff $\ell \in WFS(KB)$.*

Example 5.5 Consider KB of Example 5.4. For $I_0 = \emptyset$, we have $T_{KB}(I_0) = \emptyset$ and $U_{KB}(I_0) = \{p(a), q(a), c(a), s(a)\}$. Hence, $W_{KB}(I_0) = \{\neg p(a), \neg q(a), \neg c(a), \neg s(a)\}$ (= I_1). In the next iteration, $T_{KB}(I_1) = \{r(a)\}$ and $U_{KB}(I_1) = \{p(a), q(a), c(a), s(a)\}$. Thus, $W_{KB}(I_1) = \{\neg p(a), \neg q(a), \neg c(a), r(a), \neg s(a)\}$ (= I_2). As I_2 is total, and W_{KB} is monotonic, it follows $W_{KB}(I_2) = I_2$ and thus $lfp(W_{KB}) = I_2$. Since

$L \cup I_2$ is satisfiable, KB is w-consistent, and $WFS(KB) = \{\neg p(a), \neg q(a), \neg c(a), r(a), \neg s(a)\}$. That is, $r(a)$ is well-founded and all other atoms are unfounded relative to KB. Note that KB has the unique answer set $I = \{r(a)\}$.

Example 5.6 Consider again $KB_2 = (L_2, P_2)$ of Example 5.1. Then, it is not difficult to verify that $L \cup lfp(W_{KB_2})$ is satisfiable, and thus KB_2 is w-consistent. Furthermore, $WFS(KB_2) = lfp(W_{KB_2})$ contains in particular all facts in P_2 and all membership axioms in L_2, the concept membership axioms *electronics*(*pc_ibm*) and *product*(*pc_ibm*), and the literals *provider*(*ibm*), *similar*(*pc_ibm*, *pc_hp*), and $\neg$*offer*(pc_1). Hence, all these membership axioms and literals are consequences of KB_2 under the well-founded semantics.

6 Semantic Properties

In this section, we investigate the semantic properties (especially those relevant for the Semantic Web) of the above disjunctive dl-programs under the answer set semantics and normal dl-programs under the well-founded semantics.

6.1 Answer Set Semantics

In the ordinary case (see Section 3), every answer set of a disjunctive program P is also a minimal model of P, and the converse holds when P is positive. Intuitively, the answer set semantics of a disjunctive program P selects a set of preferred models among all minimal models of P, where the selection depends on the default negations in P. The following theorem shows that these results carry over to disjunctive dl-programs.

Theorem 6.1 *Let $KB = (L, P)$ be a disjunctive dl-program. Then, (a) every answer set of KB is a minimal model of KB, and (b) if KB is positive, then the set of all answer sets of KB is given by the set of all minimal models of KB.*

The next theorem shows that positive normal dl-programs over *DL-Lite*$_\mathcal{A}$ are either unsatisfiable or have a least model. Note that this is different from the ordinary case (where positive normal programs always have a least model), since the description logic knowledge base may make a positive normal dl-program unsatisfiable. The theorem also shows that positive normal dl-programs over *DL-Lite*$_\mathcal{A}$ have either no answer set or a unique one, which coincides with their least model. Intuitively, the answer set semantics of such dl-programs coincides with their least model semantics.

Theorem 6.2 *Let $KB = (L, P)$ be a positive normal dl-program with L in* DL-Lite$_\mathcal{A}$*. Then, (a) KB is either unsatisfiable or has a least model, and (b) KB has either no answer set or a unique one, which coincides with the least model of KB.*

An important property of integrations of rules and ontologies is that they are a faithful [50, 51] extension of both rules and ontologies. The following theorem shows that the answer set semantics of disjunctive dl-programs faithfully extends the ordinary counterpart for disjunctive programs. That is, the answer set semantics of a disjunctive dl-program with empty description logic knowledge base coincides with the ordinary answer set semantics of its disjunctive program.

Theorem 6.3 *Let $KB = (L, P)$ be a disjunctive dl-program such that $L = \emptyset$. Then, the set of all answer sets of KB coincides with the set of all ordinary answer sets of P.*

Towards faithfulness concerning the extension of description logic knowledge bases, the next theorem shows that a ground atom $a \in HB_\Phi$ is true in all answer sets of a positive disjunctive dl-program $KB = (L, P)$ iff a is true in all first-order models of $L \cup ground(P)$. The theorem and the following corollary hold also when a is a ground formula constructed from HB_Φ using $\wedge$ and $\vee$. Observe that the theorem and the following corollary do not hold for all first-order formulas a, but we actually also do not need this, *looking from the perspective of answer set programming*, since we actually cannot refer to all general first-order formulas in P.

Theorem 6.4 *Let $KB = (L, P)$ be a positive disjunctive dl-program, and let a be a ground atom from HB_Φ. Then, a is true in all answer sets of KB iff a is true in all first-order models of $L \cup ground(P)$.*

As an immediate corollary, we thus obtain that the answer set semantics of disjunctive dl-programs also faithfully extends the first-order semantics of description logic knowledge bases. That is, the answer set semantics of a disjunctive dl-program with empty disjunctive program coincides with the first-order semantics of its description logic knowledge base.

Corollary 6.1 *Let $KB = (L, P)$ be a disjunctive dl-program with $P = \emptyset$, and let $a \in HB_\Phi$. Then, a is true in all answer sets of KB iff a is true in all first-order models of L.*

It is often argued that the closed-world assumption is not very desirable in the open environment of the Semantic Web [54]. The notion of cautious reasoning from disjunctive dl-programs under the answer set semantics also has some closed-world property. However, as also shown by Theorem 6.4, this closed-world property is actually limited to the explicit use of default negations in rule bodies, and thus we can actually control very easily its use in disjunctive dl-programs.

Another aspect that may not be very desirable in the Semantic Web [54] is the *unique name assumption* (which says that any two distinct constant symbols in Φ_c represent two distinct domain objects). It turns out that we actually do not have to make this assumption, since the description logic knowledge base of a disjunctive dl-program may very well contain or imply equalities between individuals.

This result is included in the following theorem, which shows an alternative characterization of the satisfaction of L in $I \subseteq HB_\Phi$: Rather than being enlarged by a set of axioms of exponential size, L is enlarged by a set of axioms of polynomial size. This characterization essentially shows that the satisfaction of L in I corresponds to checking that

(i) the ground atoms in $I \cap DL_\Phi$ satisfy L, and

(ii) the ground atoms in $I \cap (HB_\Phi - DL_\Phi)$ do not violate any equality axiom that follows from L.

In the theorem, an equivalence relation $\sim$ on Φ_c is *admissible* with an interpretation $I \subseteq HB_\Phi$ iff $p(c_1, \ldots, c_n) \in I \Leftrightarrow p(c'_1, \ldots, c'_n) \in I$ for all n-ary predicate symbols p, where $n > 0$, and constant symbols $c_1, \ldots, c_n$, $c'_1, \ldots, c'_n \in \Phi_c$ such that $c_i \sim c'_i$ for all $i \in \{1, \ldots, n\}$.

Theorem 6.5 *Let L be a description logic knowledge base, and $I \subseteq HB_\Phi$. Then, $L \cup I \cup \{\neg b \mid b \in HB_\Phi - I\}$ is satisfiable iff $L \cup (I \cap DL_\Phi) \cup \{\neg b \mid b \in DL_\Phi - I\} \cup \{c \neq c' \mid c \not\sim c'\}$ is satisfiable for an equivalence relation $\sim$ on Φ_c admissible with I.*

The processing of conjunctive queries is important for the Semantic Web [58]. Observe that (Boolean unions of) conjunctive queries in our approach can be reduced to atomic queries. A *Boolean union of conjunctive queries* Q is of the form $\exists \boldsymbol{x}(\gamma_1(\boldsymbol{x}) \vee \cdots \vee \gamma_n(\boldsymbol{x}))$, where $\boldsymbol{x}$ is a tuple of variables, $n \geqslant 1$, and each $\gamma_i(\boldsymbol{x})$ is a conjunction of atoms constructed from predicate and constant symbols in Φ and variables in $\boldsymbol{x}$. We call Q a *conjunctive query* when $n = 1$. If we assume that $\boldsymbol{x}$ ranges over all constant symbols in Φ_c (which is sufficient for our needs, *looking from the perspective of answer set programming*, since in P we can refer only through Φ_c to elements of a first-order domain), then Q can be expressed by adding the rules $q(\boldsymbol{x}) \leftarrow \gamma_i(\boldsymbol{x})$ with $i \in \{1, \ldots, n\}$ to P and thereafter computing the set of all entailed ground instances of $q(\boldsymbol{x})$ relative to Φ_c (see also Section 7).

6.2 Well-Founded Semantics

We next explore the semantic properties of the well-founded semantics for normal dl-programs, and their relationship to the answer set semantics. As a first such property, the well-founded semantics of normal dl-programs faithfully extends the well-founded semantics of ordinary normal programs. That is, the well-founded semantics of any normal dl-program with empty description logic knowledge base coincides with the ordinary well-founded semantics of its normal program.

Theorem 6.6 *Let $KB = (L, P)$ be a normal dl-program such that $L = \emptyset$. Then, the well-founded semantics of KB coincides with the well-founded semantics of P.*

Furthermore, the well-founded semantics for normal dl-programs $KB = (L, P)$ where L is defined in *DL-Lite*$_{\mathcal{A}}$ can also be characterized in terms of the least and the greatest fixpoint of a monotonic operator γ^2_{KB} similar as the well-founded semantics for ordinary normal programs [6]. This characterization can then be used to derive further properties of the well-founded semantics for normal dl-programs. We first define the operator γ_{KB} as follows.

Definition 6.1 *For a normal dl-program $KB = (L, P)$ with L in* DL-Lite$_{\mathcal{A}}$*, the application of the operator γ_{KB} on $I \subseteq HB_\Phi$, denoted $\gamma_{KB}(I)$, is the least model of (L^+, P^I), where L^+ is defined in the same way as in Section 5.3.*

The next result shows that the operator γ_{KB} is anti-monotonic, like its counterpart for ordinary normal programs [6].

Lemma 6.1 *Let $KB = (L, P)$ be a normal dl-program with L in* DL-Lite$_{\mathcal{A}}$*. Then, γ_{KB} is anti-monotonic.*

Hence, the operator $\gamma^2_{KB}(I) = \gamma_{KB}(\gamma_{KB}(I))$, for all $I \subseteq HB_\Phi$, is monotonic, and thus has a least and a greatest fixpoint, denoted $lfp(\gamma^2_{KB})$ and $gfp(\gamma^2_{KB})$, respectively, which characterize the well-founded semantics of KB as follows.

Theorem 6.7 *Let $KB = (L, P)$ be a normal dl-program with L in* DL-Lite$_{\mathcal{A}}$*. Then, (a) KB is w-consistent iff $L \cup (lfp(\gamma^2_{KB}) \cap DL_\Phi) \cup \neg.(DL_\Phi \setminus gfp(\gamma^2_{KB}))$ is satisfiable, and (b) in that case, $a \in HB_\Phi$ is well-founded* (*resp., unfounded*) *relative to KB iff $a \in lfp(\gamma^2_{KB})$* (*resp., $a \notin gfp(\gamma^2_{KB})$*).

The following theorem shows that for normal dl-programs, consistency under the answer set semantics implies consistency under the well-founded semantics. The converse, however, does not hold in general, unless the well-founded semantics is defined and total (that is, two-valued, which means that it contains

either a or $\neg a$ for every $a \in HB_\Phi$) as, for example, in the positive normal case. This is due to the fact that it may not always be possible to complete the partial model of the well-founded semantics to a total model.

Theorem 6.8 *Let $KB = (L, P)$ be a normal dl-program. If KB is consistent, then KB is w-consistent.*

The next theorem shows that the well-founded semantics for normal dl-programs approximates their answer set semantics. That is, every well-founded ground atom is true in all answer sets, and every unfounded one is false in all answer sets.

Theorem 6.9 *Let $KB = (L, P)$ be a consistent normal dl-program with L in* DL-Lite$_\mathcal{A}$*. Then, every answer set of KB includes all atoms $a \in HB_\Phi$ that are well-founded relative to KB and no atom $a \in HB_\Phi$ that is unfounded relative to KB.*

Recall that a ground atom a is a cautious (resp., brave) consequence under the answer set semantics of a normal dl-program KB iff a is true in every (resp., some) answer set of KB. Hence, as a corollary of Theorem 6.9, under the answer set semantics, every well-founded and no unfounded ground atom is a cautious (resp., brave) consequence.

Corollary 6.2 *Let $KB = (L, P)$ be a consistent normal dl-program with L in* DL-Lite$_\mathcal{A}$*. Then, under the answer set semantics, every well-founded atom $a \in HB_\Phi$ relative to KB is a cautious consequence of KB, and no unfounded atom $a \in HB_\Phi$ relative to KB is a brave consequence of KB.*

The following theorem shows that if the well-founded semantics of a normal dl-program is total, then it specifies the only answer set of such a dl-program.

Theorem 6.10 *Let $KB = (L, P)$ be a consistent normal dl-program with L in* DL-Lite$_\mathcal{A}$*. If every $a \in HB_\Phi$ is either well-founded or unfounded w.r.t. KB, then the set of all well-founded $a \in HB_\Phi$ w.r.t. KB is the only answer set of KB.*

Like in the case of ordinary normal programs, the well-founded semantics for satisfiable positive normal dl-programs is total and coincides with their least model semantics. This result can be elegantly proved using the characterization of the well-founded semantics given in terms of γ^2_{KB}.

Theorem 6.11 *Let $KB = (L, P)$ be a consistent positive normal dl-program with L in* DL-Lite$_\mathcal{A}$*. Then, (a) $WFS(KB)$ is total, that is, $WFS(KB)^+ \cup (\neg.WFS(KB))^+ = HB_\Phi$, and (b) $WFS(KB) \cap HB_\Phi$ is the least model of KB, which coincides with the unique answer set of KB.*

7 Algorithms

In this section, we describe algorithms for deciding whether a disjunctive dl-program has an answer set, and for deciding brave and cautious consequences of ground atoms from disjunctive dl-programs under the answer set semantics. Furthermore, we provide algorithms for deciding whether a normal dl-program has a well-founded semantics, and for deciding entailment of ground literals from normal dl-programs under the well-founded semantics.

Algorithm consistency

Input: vocabulary Φ, disjunctive dl-program $KB = (L, P)$.
Output: *Yes*, if KB has an answer set; *No*, otherwise.

1. **if** there exists $I \subseteq HB_\Phi$ such that I is a minimal
2. model of $KB^I = (L, P^I)$ **then return** *Yes*
3. **else return** *No*.

Figure 1: Algorithm consistency.

7.1 Answer Set Semantics

The problem of deciding whether a disjunctive dl-program $KB = (L, P)$ has an answer set can be solved by a simple guess-and-check algorithm, which guesses a subset I of the finite Herbrand base HB_Φ, computes the FLP-reduct $KB^I = (L, P^I)$, and then checks that I is in fact a minimal model of KB^I (see Fig. 1).

The problem of deciding brave and cautious consequences of ground atoms from disjunctive dl-programs under the answer set semantics can be reduced to deciding answer set existence (like in the ordinary case), since a ground atom $a \in HB_\Phi$ is true in some (resp., every) answer set of a disjunctive dl-program $KB = (L, P)$ iff $(L, P \cup \{\leftarrow not\ a\})$ (resp., $(L, P \cup \{\leftarrow a\})$) has an (resp., no) answer set.

7.2 Well-Founded Semantics

By Theorem 6.7, deciding whether the well-founded semantics exists for a normal dl-program KB over *DL-Lite*$_\mathcal{A}$ and eventually computing it can be done by two finite fixpoint iterations, via γ_{KB}, using in turn finite fixpoint iterations for computing the least models of positive normal dl-programs, via their immediate consequence operator. Then, entailment of ground literals ℓ from KB under the well-founded semantics can be decided by checking whether $\ell \in WFS(KB)$. By Theorem 6.9, $WFS(KB)$ can also be used to speed up the guess-and-check algorithm for deciding whether KB has an answer set.

More specifically, for any positive normal dl-program $KB = (L, P)$ with L in *DL-Lite*$_\mathcal{A}$, the least model of KB, if it exists, coincides with the least fixpoint of the immediate consequence operator T_{KB}, denoted $lfp(T_{KB})$, which is defined as follows for every $I \subseteq HB_\Phi$:

$$T_{KB}(I) = \{H(r) \mid r \in ground(P),\ I \models b \text{ for all } b \in B(r)\} \cup \{a \in DL_\Phi \mid L \cup (I \cap DL_\Phi) \models a\}\,.$$

In order to compute the well-founded semantics of a normal dl-program $KB = (L, P)$ with L in *DL-Lite*$_\mathcal{A}$, that is, by Theorem 6.7, $WFS(KB) = lfp(\gamma^2_{KB}) \cup \neg.(HB_\Phi \setminus gfp(\gamma^2_{KB}))$, if it exists, we compute the least and the greatest fixpoint of γ^2_{KB} as the limits of the two fixpoint iterations

$$lfp(\gamma^2_{KB}) = U_\infty = \textstyle\bigcup_{i \geq 0} U_i, \text{ where } U_0 = \emptyset, \text{ and } U_{i+1} = \gamma^2_{KB}(U_i), \text{ for } i \geq 0, \text{ and}$$
$$gfp(\gamma^2_{KB}) = O_\infty = \textstyle\bigcap_{i \geq 0} O_i, \text{where } O_0 = HB_\Phi, \text{ and } O_{i+1} = \gamma^2_{KB}(O_i), \text{ for } i \geq 0,$$

respectively, which are both reached within $|HB_\Phi|$ many steps. Recall that the application of the operator γ_{KB} on $I \subseteq HB_\Phi$, denoted $\gamma_{KB}(I)$, is the least model of $KB^{I,+} = (L^+, P^I)$, where L^+ is defined as in Section 5.3. As argued above, $\gamma_{KB}(I)$ coincides with $lfp(T_{KB^{I,+}})$, for all $I \subseteq HB_\Phi$. To compute $\gamma_{KB}(I)$, for all $I \subseteq HB_\Phi$, we thus compute the least fixpoint of $T_{KB^{I,+}}$ as the limit of the fixpoint iteration

$$lfp(T_{KB^{I,+}}) = S_\infty = \textstyle\bigcup_{i \geq 0} S_i, \text{ where } S_0 = \emptyset, \text{ and } S_{i+1} = T_{KB^{I,+}}(S_i), \text{ for } i \geq 0,$$

which is also reached within $|HB_\Phi|$ many steps.

By Theorem 6.7, it then holds that (a) $WFS(KB)$ exists iff $L \cup (\mathit{lfp}(\gamma^2_{KB}) \cap DL_\Phi) \cup \neg.(DL_\Phi \setminus \mathit{gfp}(\gamma^2_{KB}))$ is satisfiable, and (b) $\ell \in WFS(KB)$ iff $\ell \in \mathit{lfp}(\gamma^2_{KB}) \cup \neg.(HB_\Phi \setminus \mathit{gfp}(\gamma^2_{KB}))$.

8 Complexity

In this section, we give a precise picture of the complexity of deciding whether a disjunctive dl-program has an answer set, and of deciding brave and cautious consequences of ground atoms from disjunctive dl-programs under the answer set semantics. We also give a precise picture of the complexity of deciding whether a normal dl-program has a well-founded semantics, and of deciding entailment of ground literals from normal dl-programs under the well-founded semantics.

8.1 Complexity Classes

We assume that the reader has some elementary background in complexity theory (see, e.g., [38, 53]). We now briefly recall the complexity classes that we encounter in the complexity results below. The class NP contains all decision problems that can be solved in polynomial time on a nondeterministic Turing machine, while EXP (resp., NEXP) contains all decision problems solvable in exponential time on a deterministic (resp., nondeterministic) Turing machine. The class $\mathrm{NEXP}^{\mathrm{NP}}$ contains all problems that are decidable in exponential time on a nondeterministic Turing machine with the help of an NP oracle, while $\mathrm{co\text{-}NEXP}^{\mathrm{NP}}$ is the complementary class of $\mathrm{NEXP}^{\mathrm{NP}}$, which has yes- and no-instances interchanged.

8.2 Answer Set Semantics

We now show that the problems of deciding consistency and brave / cautious consequences have the same complexity in disjunctive dl-programs under the answer set semantics as in ordinary disjunctive programs under the answer set semantics.

The following theorem shows that deciding the consistency of disjunctive dl-programs is complete for $\mathrm{NEXP}^{\mathrm{NP}}$. The lower bound follows from the $\mathrm{NEXP}^{\mathrm{NP}}$-hardness of deciding the consistency of ordinary disjunctive programs [15]. The upper bound follows from the result that deciding knowledge base satisfiability in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$) is complete for EXP (resp., NEXP) [61, 33].

Theorem 8.1 *Given Φ and a disjunctive dl-program $KB = (L, P)$ with L in $\mathcal{SHIF}(\mathbf{D})$ or $\mathcal{SHOIN}(\mathbf{D})$, deciding whether KB has an answer set is complete for* $\mathrm{NEXP}^{\mathrm{NP}}$.

The next theorem shows that deciding cautious (resp., brave) consequences of ground atoms from disjunctive dl-programs is complete for $\mathrm{co\text{-}NEXP}^{\mathrm{NP}}$ (resp., $\mathrm{NEXP}^{\mathrm{NP}}$). This follows from Theorem 8.1, since consistency checking and cautious (resp., brave) reasoning can be reduced to each other.

Theorem 8.2 *Given Φ, a disjunctive dl-program $KB = (L, P)$ with L in $\mathcal{SHIF}(\mathbf{D})$ or $\mathcal{SHOIN}(\mathbf{D})$, and $a \in HB_\Phi$, deciding whether a holds in every (resp., some) answer set of KB is complete for* $\mathrm{co\text{-}NEXP}^{\mathrm{NP}}$ *(resp.,* $\mathrm{NEXP}^{\mathrm{NP}}$*).*

8.3 Well-Founded Semantics

We next show that the main computational problems for normal dl-programs under the well-founded semantics have the same complexity as the ones for ordinary normal programs under the well-founded semantics.

The following theorem shows that deciding consistency and entailment of ground literals for normal dl-programs over *DL-Lite*$_{\mathcal{A}}$ under the well-founded semantics are complete for EXP. Hardness for EXP follows from the hardness for EXP of deciding, given an ordinary positive program P and a ground atom a, whether P logically entails a [15]. Membership in EXP follows from the fact that (a) computing the well-founded semantics of ordinary normal programs can be done in exponential time, and (b) instance checking and knowledge base satisfiability in *DL-Lite*$_{\mathcal{A}}$ can be done in polynomial time.

Theorem 8.3 *(a) Given a vocabulary Φ and a normal dl-program $KB = (L, P)$ with L in* DL-Lite$_{\mathcal{A}}$*, deciding whether KB is w-consistent is* EXP*-complete. (b) Given additionally a ground literal ℓ, deciding whether ℓ is a consequence of KB under the well-founded semantics is* EXP*-complete.*

9 Data Tractability

We now show that deciding consistency and entailment of ground literals for normal dl-programs under the well-founded semantics can be done in polynomial time in the data complexity. Furthermore, we delineate a special case where this can even be done in LOGSPACE in the data complexity.

9.1 Polynomial Case

The following theorem shows that deciding consistency resp. entailment of ground literals for normal dl-programs over *DL-Lite*$_{\mathcal{A}}$ under the well-founded semantics are complete for P in the data complexity. Recall that the complexity class P contains all decision problems that can be solved in polynomial time on a deterministic Turing machine, and that the data complexity describes the case where all but the facts and the concept and role membership axioms in dl-programs are fixed. The bounds in the theorem follow from a similar argumentation as in the general case, except that now, deciding, given an ordinary positive program P and a ground atom a, whether P logically entails a is hard for P in the data complexity [15], and computing the well-founded semantics of ordinary normal programs is in P in the data complexity. This data tractability result for deciding consistency and entailment of ground literals nicely generalizes the data tractability result presented in the ESWC-2007 abstract of this paper.

Theorem 9.1 *(a) Given a vocabulary Φ and a normal dl-program $KB = (L, P)$ with L in* DL-Lite$_{\mathcal{A}}$*, deciding whether KB is w-consistent is P-complete in the data complexity. (b) Given additionally a ground literal ℓ, deciding whether ℓ is a consequence of KB under the well-founded semantics is P-complete in the data complexity.*

9.2 First-Order Rewritable Case

We now show that deciding consistency and entailment of ground literals for normal dl-programs $KB = (L, P)$ with L in *DL-Lite*$_{\mathcal{A}}$ under the well-founded semantics are even first-order rewritable, and thus can be done in LOGSPACE in the data complexity, when P is acyclic. Hence, deciding consistency and entailment of ground literals for such KB under the well-founded semantics can be done very efficiently by commercial, SQL-expressive relational database systems.

We first formalize the notion of first-order rewritability for the consistency and literal entailment problems in normal dl-programs under the well-founded semantics. The w-consistency problem in (resp., entailment problem of a ground literal ℓ from w-consistent) normal dl-programs $KB = (L, P)$ is *first-order rewritable* iff it can be expressed in terms of a first-order formula ϕ over the set F of all concept, role, and attribute membership axioms in L and all facts in P, that is, KB is w-consistent (resp., $\ell \in WFS(KB)$) iff $I_F \models \phi$, where I_F is the total Herbrand interpretation satisfying exactly F.

We next define the notion of acyclicity for ordinary normal programs and normal dl-programs as follows. Given a normal program P, we denote by $\mathcal{P}_P$ the set of all predicate symbols in P. We say P is *acyclic* iff a mapping $\kappa\colon \mathcal{P}_P \to \{0, 1, \ldots, n\}$ exists such that for every $r \in P$, the predicate symbol p of $H(r)$, and every predicate symbol q of some $b \in B(r)$, it holds that $\kappa(p) > \kappa(q)$. A normal dl-program $KB = (L, P)$ is *acyclic* iff

(i) P is acyclic, and

(ii) L can be partitioned into description logic knowledge bases $L_1^I, \ldots, L_m^I, L^O$ in *DL-Lite*$_\mathcal{A}$ over pairwise disjoint sets of atomic concepts, atomic roles, and attributes such that the atomic concepts, atomic roles, and attributes of $L_1^I, \ldots, L_m^I$ only occur in bodies of rules in P and the ones of L^O only occur in heads of rules in P.

Intuitively, acyclic normal dl-programs $KB = (L, P)$ allow for reading out instances of concepts, roles, and attributes from several input ontologies $L_1^I, \ldots, L_m^I$, elaborating them in an acyclic normal program P, and then merging the result into an output ontology L^O.

The following theorem shows that the two problems of deciding consistency and of deciding entailment of ground literals for acyclic normal dl-programs under the well-founded semantics are both first-order rewritable (and thus can be done in LOGSPACE in the data complexity).

Theorem 9.2 *(a) Given an alphabet Φ and an acyclic normal dl-program $KB = (L, P)$ with L in* DL-Lite$_\mathcal{A}$*, deciding whether KB is w-consistent is first-order rewritable. (b) Given additionally a ground literal ℓ, deciding whether KB entails ℓ under the well-founded semantics is first-order rewritable.*

10 Related Work

There is a large body of related works on combining rules and ontologies, which can essentially be divided into the following three lines of research: (a) loose integration of rules and ontologies, (b) tight integration of rules and ontologies, and (c) reductions from description logics to logic programming formalisms. Further related works deal with such combinations under the well-founded semantics and with the more general use of the well-founded semantics in the context of the Web. In this section, we discuss only the works that are most closely related to the framework of this paper.

Representatives of the loose integration of rules and ontologies are in particular the dl-programs in [25, 21], their extension to HEX-programs [23, 24], to probabilistic dl-programs [45, 46, 47], and to fuzzy dl-programs [43, 44]. The combination of defeasible reasoning with description logics in [3], the calls to description logic reasoners in TRIPLE [59], and the hybrid MKNF knowledge bases in [50, 51] are also close in spirit. More concretely, compared to the present paper, the dl-programs $KB = (L, P)$ in [25, 21] also consist of a description logic knowledge base L and a normal program P. However, P may also contain classical negations, and rather than using concepts and roles from L as predicates in P, rule bodies in P may only contain queries to L, which may also contain facts as additional input to L. Like in this paper, P is

interpreted relative to Herbrand interpretations under the answer set semantics, while L is interpreted relative to first-order interpretations under the classical model-theoretic semantics. However, differently from the concepts and roles in P here, the queries in P in [25, 21] are evaluated independently from each other. HEX-programs [23, 24] extend the approach to dl-programs in [25, 21] by multiple sources of external knowledge, with possibly different semantics, while probabilistic dl-programs [45, 46, 47] and fuzzy dl-programs [43, 44] are extensions by probabilistic uncertainty and fuzzy vagueness, respectively. Closely related to the dl-programs in [25, 21] are also the hybrid MKNF knowledge bases in [50, 51]. They essentially allow for querying a description logic knowledge base L via the operators $\mathbf{K}$ and $\mathbf{not}$, which can be used more flexibly than the queries in [25, 21] (the operators can also occur in rule heads, while the queries are restricted to rule bodies), but which do not allow for passing facts to L in the form of query arguments. Note that closely related to the hybrid MKNF knowledge bases in [50, 51] is also the embedding of non-ground logic programs into autoepistemic logic in [16]. Recall that Example 2.3 shows that our novel dl-programs here generally do not have the same meaning as the dl-programs in [25, 21] (note that a similar example can be constructed for the approach in [50, 51]).

Some representatives of tight integrations of rules and ontologies are in particular the works due to Donini et al. [18], Levy and Rousset [42], Grosof et al. [30], Motik et al. [52], Heymans et al. [31], and Rosati [56, 58]. SWRL [34] and WRL [2] also belong to this category. Closest in spirit to this paper among the above works is perhaps Rosati's approach [56, 58]. Like here, Rosati's hybrid knowledge bases also consist of a description logic knowledge base L and a disjunctive program (with default negations) P, where concepts and roles in L may act as predicate symbols in P. However, differently from this paper, Rosati partitions the predicates of L and P into description logic predicates and logic program predicates, where the former are interpreted under the classical model-theoretic semantics, while the latter are interpreted under the answer set semantics (and thus in particular default negations of concepts and roles are not allowed in P). Furthermore, differently from this paper, he also assumes a syntactic restriction on rules (called *weak safety*) to gain decidability, and he assumes the standard names assumption, which includes the unique name assumption.

The works reducing description logics to logic programming are less closely related to the present paper. Some representatives are in particular the ones by Alsaç and Baral [1], Swift [60], Heymans and Vermeir [32], and Motik et al. [37].

For several of the above combinations of rules and ontologies, a well-founded semantics has been defined; more specifically, the works [26], [40], and [20] define a well-founded semantics for the loosely integrated dl-programs in [25, 21], for the hybrid MKNF knowledge bases in [50, 51], and for an integration of rules and ontologies that is close in spirit to Rosati's approach [56, 58], respectively. As for the more general use of the well-founded semantics in the context of the Web, several reasoners adopt it for handling nonmonotonic negation, including $\mathcal{F}$lora-2[1] (which builds on XSB[2]) and OntoBroker[3] that are based on F-Logic [39], and IRIS and MINS,[4] towards the WSML-Rule language [17]. Here, F-Logic is a formal model for a deductive object-oriented database system, which combines the structural aspects of object-oriented and frame-based languages (and which uses in particular rules to define ontological knowledge), while WSML (Web Service Modeling Language) is a formal language for the specification of different aspects of Semantic Web Services, with WSML-Rule being a logic programming extension of Grosof et al.'s DLP fragment [30]. Hence, both F-Logic and WSML-Rule are less closely related to the tight disjunctive

[1] http://flora.sourceforge.net/

[2] http://xsb.sourceforge.net/

[3] http://www.ontoprise.de/en/home/products/ontobroker/

[4] http://iris-reasoner.org/, http://tools.sti-innsbruck.at/mins/

dl-programs introduced and explored in this paper.

11 Conclusion

We have presented a novel combination of disjunctive logic programs under the answer set semantics with description logics for the Semantic Web. The combination is based on a well-balanced interface between disjunctive logic programs and description logics, which guarantees the decidability of the resulting formalism without assuming any syntactic restrictions on the resulting language (such as syntactic safety conditions and/or syntactic partitionings of the vocabulary). We have shown that the new formalism has very nice semantic properties. In particular, it faithfully extends both disjunctive logic programs under the answer set semantics and description logics under the standard first-order semantics. We have also provided algorithms and precise complexity results for the new formalism. Furthermore, we have defined the well-founded semantics for the special case of normal dl-programs, and explored its semantic and computational properties. In particular, we have shown that the well-founded semantics faithfully extends its classical counterpart, and that it approximates the answer set semantics. We have also described algorithms for consistency checking and literal entailment under the well-founded semantics, and we have analyzed the data and general complexity of these two central computational problems. As a crucial property, normal dl-programs under the well-founded semantics allow for tractable consistency checking and for tractable literal entailment in the data complexity, and they have even a first-order rewritable (and thus LOGSPACE data complexity) special case, which is especially interesting for representing (deterministic) ontology mappings.

Note that the results of this paper are not restricted to the expressive description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$ and to the tractable description logic *DL-Lite*$_\mathcal{A}$ as underlying ontology languages. In particular, the results for the well-founded semantics also hold when any other tractable description logic from the *DL-Lite* family [14] is used instead.

Conceptually, the introduced tightly integrated disjunctive dl-programs are a quite natural combination of ordinary disjunctive programs and description logic knowledge bases under their standard semantics, without imposing syntactic restrictions, and with a clear interface via common predicates. Other approaches to such combinations often have syntactic restrictions and/or semantic drawbacks. For these conceptual reasons, we can expect the new disjunctive dl-programs to be quite usable for target end users. Furthermore, we can expect them to have nice features concerning composition, integration of knowledge, and discovery and verification issues. Like the standard toolsets, algorithms, and complexity results of disjunctive programs, we can expect these features to similarly carry over to disjunctive dl-programs from their ordinary counterparts. For example, in a companion paper [13], it has been shown that the new disjunctive dl-programs and a probabilistic generalization thereof can nicely be used to represent and reason with exact and uncertain ontology mappings.

The presented mechanism of integrating rules and ontologies is of general importance, since it can actually also be used for the decidable integration of other reasoning techniques (such as reasoning about defaults, probabilistic uncertainty, and fuzzy vagueness) with description logics, since it applies to all reasoning techniques that are based on interpretations over finite Herbrand bases (or also finite sets of propositional symbols). It thus paves the way for decidable reasoning formalisms on top of description logics for the Semantic Web. The collections of companion papers [48, 49] and [10, 13, 12] explore the use of this novel integration in fuzzy and in probabilistic description logic programs, respectively.

We leave for future work the implementation of tightly integrated disjunctive and normal dl-programs. It would also be interesting to explore whether the first-order rewritability result can be extended to an even

larger class of tightly integrated normal dl-programs. Another interesting issue is to extend the presented approaches to disjunctive and normal dl-programs by classical negation and by functions, if possible.

Appendix A: Proofs for Section 5

Proof of Lemma 5.1. (a) ($\Rightarrow$) Suppose $L \cup \{\neg C(a)\}$ is satisfiable. Let $\mathcal{I}$ be any first-order model of $L \cup \{\neg C(a)\}$. Let $\mathcal{I}$ be extended to $\mathcal{I}'$ by $B^{\mathcal{I}'} = \{a^{\mathcal{I}}\}$. Then, $\mathcal{I}'$ is a model of $L' = L \cup \{B(a), B \sqsubseteq \neg C\}$. That is, L' is satisfiable.

($\Leftarrow$) Suppose $L \cup \{B(a), B \sqsubseteq \neg C\}$ is satisfiable. Let $\mathcal{I}$ be any first-order model of $L \cup \{B(a), B \sqsubseteq \neg C\}$. Hence, the restriction of $\mathcal{I}$ to the original vocabulary without B is a model of $L \cup \{\neg C(a)\}$. That is, $L \cup \{\neg C(a)\}$ is satisfiable.

(b) ($\Rightarrow$) Suppose $L \cup \{\neg R(b, c)\}$ is satisfiable. Let $\mathcal{I}$ be any first-order model of $L \cup \{\neg R(b, c)\}$. Let $\mathcal{I}$ be extended to $\mathcal{I}'$ by $B^{\mathcal{I}'} = \{b^{\mathcal{I}}\}$ and $C^{\mathcal{I}'} = \{c^{\mathcal{I}}\}$. Then, $\mathcal{I}'$ is a model of $L \cup \{B(b), C(c)\}$. Since $\mathcal{I}$ satisfies $\neg R(b, c)$, also $\mathcal{I}'$ satisfies $\neg R(b, c)$. Thus, $\mathcal{I}'$ is a model of $L' = L \cup \{B(b), C(c), \exists R.C \sqsubseteq \neg B\}$. That is, L' is satisfiable.

($\Leftarrow$) Suppose $L \cup \{B(b), C(c), \exists R.C \sqsubseteq \neg B\}$ is satisfiable. Let $\mathcal{I}$ be any first-order model of $L \cup \{B(b), C(c), \exists R.C \sqsubseteq \neg B\}$. Observe that $\mathcal{I}$ is a model of $\neg R(b, c)$, since otherwise it would not satisfy $\exists R.C \sqsubseteq \neg B$. Hence, the restriction of $\mathcal{I}$ to the original vocabulary without B and C is a model of $L \cup \{\neg R(b, c)\}$. That is, $L \cup \{\neg R(b, c)\}$ is satisfiable. □

Proof of Lemma 5.2. Suppose $U_1, U_2 \subseteq HB_\Phi$ are both unfounded sets of KB relative to I. We now show that $(*)$ holds for $U = U_1 \cup U_2$. Consider any $a \in U_1$. Then, (a) and (b) hold for $U = U_1$, and thus (a) and (b) hold for $U = U_1 \cup U_2$. Similarly, for any $a \in U_2$, (a) and (b) hold for $U = U_1 \cup U_2$. In summary, for any $a \in U_1 \cup U_2$, (a) and (b) hold for $U = U_1 \cup U_2$. That is, $(*)$ holds for $U = U_1 \cup U_2$. □

Proof of Lemma 5.3. It is sufficient to show that T_{KB} and U_{KB} are monotonic. Let $J_1 \subseteq J_2 \subseteq Lit_\Phi$ be consistent. We first show that T_{KB} is monotonic. If (a) or (b) in the definition of T_{KB} hold for $I = J_1$, then they also hold for $I = J_2$. That is, $T_{KB}(J_1) \subseteq T_{KB}(J_2)$. We next prove that U_{KB} is monotonic. If $(*)$ holds for $I = J_1$, then $(*)$ holds for $I = J_2$. Hence, every unfounded set of KB relative to J_1 is also an unfounded set of KB relative to J_2. Thus, $U_{KB}(J_1) \subseteq U_{KB}(J_2)$. □

Appendix B: Proofs for Section 6

Proof of Theorem 6.1. (a) Let $I \subseteq HB_\Phi$ be any answer set of KB. That is, I is a minimal model of $KB^I = (L, P^I)$. In particular, (i) $I \models L$ and (ii) $I \models r$ for every $r \in P^I$. That is, (i) $I \models L$ and (ii) $I \models r$ for every $r \in ground(P)$ with $I \models B(r)$. This is equivalent to (i) $I \models L$ and (ii) $I \models r$ for every $r \in ground(P)$. That is, I is a model of KB. We now show that I is also a minimal model of KB. Towards a contradiction, suppose that there exists a model $J \subset I$ of KB. That is, (i) $J \models L$ and (ii) $J \models r$ for every $r \in ground(P)$. In particular, (i) $J \models L$ and (ii) $J \models r$ for every $r \in ground(P)$ with $I \models B(r)$. That is, J is also a model of KB^I. But this contradicts I being a minimal model of KB^I. In summary, this shows that I is a minimal model of KB.

(b) Let $I \subseteq HB_\Phi$ be any minimal model of the positive disjunctive dl-program $KB = (L, P)$. Hence, (i) $I \models L$ and (ii) $I \models r$ for every $r \in ground(P)$. In particular, (i) $I \models L$ and (ii) $I \models r$ for every $r \in ground(P)$

with $I \models B(r)$. That is, I is a model of $KB^I = (L, P^I)$. We now show that I is also a minimal model of KB^I. Towards a contradiction, suppose that there exists a model $J \subset I$ of KB^I. That is, (i) $J \models L$ and (ii) $J \models r$ for every $r \in ground(P)$ with $I \models B(r)$. Since P is positive and $J \subset I$, it follows that $I \not\models B(r)$ implies $J \not\models B(r)$. So, (i) $J \models L$ and (ii) $J \models r$ for every $r \in ground(P)$. That is, J is a model of KB. But this contradicts I being a minimal model of KB. In summary, I is a minimal model of KB^I. That is, I is an answer set of KB. □

Proof of Theorem 6.2. (a) Suppose that KB is satisfiable. Let the interpretation $I \subseteq HB_\Phi$ be the set of all $a \in HB_\Phi$ logically entailed by $L \cup ground(P)$. Then, I is a least model of $KB = (L, P)$.

(b) Suppose that KB has some answer set. By Theorem 6.1, the set of all answer sets is given by the set of all minimal models of KB. As L is defined in *DL-Lite*$_\mathcal{A}$, there exists a least model of KB, which is thus the only answer set of KB. □

Proof of Theorem 6.3. Observe first that $I \subseteq HB_\Phi$ is a model of $KB^I = (L, P^I)$ iff (i) $I \models L$ and (ii) $I \models r$ for every $r \in P^I$. Since $L = \emptyset$, this is equivalent to $I \models r$ for every $r \in P^I$. Thus, $I \subseteq HB_\Phi$ is a minimal model of KB^I iff I is a minimal model of P^I. That is, $I \subseteq HB_\Phi$ is an answer set of KB iff I is an ordinary answer set of P. □

Proof of Theorem 6.4. Observe first that, by Theorem 6.1, since P is positive, the set of all answer sets of KB is given by the set of all minimal models $I \subseteq HB_\Phi$ of KB. Observe then that $a \in HB_\Phi$ is true in all minimal models $I \subseteq HB_\Phi$ of KB iff a is true in all models $I \subseteq HB_\Phi$ of KB. It thus remains to show that a is true in all models $I \subseteq HB_\Phi$ of KB iff a is true in all first-order models of $L \cup ground(P)$:

($\Rightarrow$) Suppose that a is true in all models $I \subseteq HB_\Phi$ of KB. Let $\mathcal{I}$ be any first-order model of $L \cup ground(P)$. Let $I \subseteq HB_\Phi$ be defined by $b \in I$ iff $\mathcal{I} \models b$. Then, $\mathcal{I}$ is a model of $L^\star = L \cup I \cup \{\neg b \mid b \in HB_\Phi - I\}$, and thus $L^\star$ is satisfiable. Hence, I is a model of L. Since $\mathcal{I}$ is a model of $ground(P)$, also I is a model of $ground(P)$. In summary, this shows that I is a model of KB. Hence, a is true in I, and thus a is true in $\mathcal{I}$. Overall, this shows that a is true in all first-order models of $L \cup ground(P)$.

($\Leftarrow$) Suppose that a is true in all first-order models of $L \cup ground(P)$. Let $I \subseteq HB_\Phi$ be any model of KB. Then, $L^\star = L \cup I \cup \{\neg b \mid b \in HB_\Phi - I\}$ is satisfiable. Let $\mathcal{I}$ be a first-order model of $L^\star$. Then, $\mathcal{I}$ is in particular a model of L. Furthermore, since I is a model of $ground(P)$, also $\mathcal{I}$ is a model of $ground(P)$. In summary, $\mathcal{I}$ is a model of $L \cup ground(P)$. It thus follows that a is true in $\mathcal{I}$, and thus a is also true in I. Overall, this shows that a is true in all models $I \subseteq HB_\Phi$ of KB. □

Proof of Theorem 6.5. ($\Rightarrow$) Let $\mathcal{I}$ be a first-order model of $L^\star = L \cup I \cup \{\neg b \mid b \in HB_\Phi - I\}$. Let the equivalence relation $\sim$ on Φ_c be defined by $c \sim d$ iff $c^\mathcal{I} = d^\mathcal{I}$. Since $\mathcal{I}$ is a model of $L^\star$, it follows that $\sim$ is admissible with I. Furthermore, it follows that $\mathcal{I}$ is a model of $L \cup (I \cap DL_\Phi) \cup \{\neg b \mid b \in DL_\Phi - I\} \cup \{c \neq c' \mid c \not\sim c'\}$.

($\Leftarrow$) Let $\mathcal{I}$ be a model of $L \cup (I \cap DL_\Phi) \cup \{\neg b \mid b \in DL_\Phi - I\} \cup \{c \neq c' \mid c \not\sim c'\}$ for some $\sim$ admissible with I. Thus, $\mathcal{I}$ can be extended to a model $\mathcal{I}'$ of $L \cup I \cup \{\neg b \mid b \in HB_\Phi - I\}$ by $\mathcal{I}' \models b$ iff $b \in I$, for all $b \in HB_\Phi - DL_\Phi$. □

Proof of Theorem 6.6. Immediate by the observation that (i) the notion of unfounded set and the operator U_{KB} for normal dl-programs $KB = (\emptyset, P)$ coincide with the notion of unfounded set and the operator U_P for the ordinary normal program P, respectively, and (ii) the operators T_{KB} and W_{KB} for normal dl-programs

$KB = (\emptyset, P)$ have the same fixpoints as the operators T_P and W_P for the ordinary normal program P, respectively. In particular, for such normal dl-programs, $L \cup \mathit{lfp}(W_{KB}) = \mathit{lfp}(W_P)$ is always satisfiable. □

Proof of Lemma 6.1. Let $I \subseteq J \subseteq HB_\Phi$. Then, $P^J \subseteq P^I$. Hence, every model of (L^+, P^I) is also a model of (L^+, P^J). Thus, the least model of (L^+, P^J) is a subset of every model of (L^+, P^I), and thus in particular also of the least model of (L^+, P^I). That is, γ_{KB} is anti-monotonic. □

Proof of Theorem 6.7. Let $W = \mathit{lfp}(W_{KB})$ and $W^- = HB_\Phi \setminus \neg.W$. Observe then that $W^+ \subseteq W^-$. We first show that (i) W^+ and W^- are fixpoints of γ^2_{KB}, which implies that $\mathit{lfp}(\gamma^2_{KB}) \subseteq W^+ \subseteq W^- \subseteq \mathit{gfp}(\gamma^2_{KB})$. We then show that (ii) $I = \mathit{lfp}(\gamma^2_{KB}) \cup \neg.(HB_\Phi \setminus \mathit{gfp}(\gamma^2_{KB}))$ is a fixpoint of W_{KB}, which implies that $W \subseteq I$, and thus $W^+ \subseteq \mathit{lfp}(\gamma^2_{KB})$ and $\mathit{gfp}(\gamma^2_{KB}) \subseteq W^-$. This then shows that $W^+ = \mathit{lfp}(\gamma^2_{KB})$ and $W^- = \mathit{gfp}(\gamma^2_{KB})$, and consequently (a) KB is w-consistent iff $L \cup (\mathit{lfp}(\gamma^2_{KB}) \cap DL_\Phi) \cup \neg.(DL_\Phi \setminus \mathit{gfp}(\gamma^2_{KB}))$ is satisfiable, and (b) in that case, $a \in HB_\Phi$ is well-founded (resp., unfounded) relative KB iff $a \in \mathit{lfp}(\gamma^2_{KB})$ (resp., $a \notin \mathit{gfp}(\gamma^2_{KB})$).

As for (i), it is not difficult to verify that $\gamma_{KB}(W^+) = W^-$ and $\gamma_{KB}(W^-) = W^+$, which then immediately implies that W^+ and W^- are fixpoints of γ^2_{KB}.

As for (ii), it is not difficult to verify that $\gamma_{KB}(\mathit{lfp}(\gamma^2_{KB})) = \mathit{gfp}(\gamma^2_{KB})$ and $\gamma_{KB}(\mathit{gfp}(\gamma^2_{KB})) = \mathit{lfp}(\gamma^2_{KB})$, which implies that $\neg.U_{KB}(I) = I \setminus I^+$ and $T_{KB}(I) = I^+$. That is, $W_{KB}(I) = I$. □

Proof of Theorem 6.8. Suppose $KB = (L, P)$ has an answer set. That is, there exists an interpretation $I \subseteq HB_\Phi$ that is a minimal model of $KB^I = (L, P^I)$. Here, I is a model of KB^I iff $L \cup I \cup \neg.(HB_\Phi \setminus I)$ is satisfiable and I is a model of P^I. Observe then that $I \cup \neg.(HB_\Phi \setminus I)$ is a fixpoint of W_{KB}, and thus $\mathit{lfp}(W_{KB}) \subseteq I \cup \neg.(HB_\Phi \setminus I)$. Since $L \cup I \cup \neg.(HB_\Phi \setminus I)$ is satisfiable, also $L \cup \mathit{lfp}(W_{KB})$ is satisfiable. This shows that the well-founded semantics of KB exists. □

Proof of Theorem 6.9. For any $I \subseteq HB_\Phi$, if I is an answer set of KB, then I is a fixpoint of γ_{KB} and thus of γ^2_{KB}. Since $\mathit{lfp}(\gamma^2_{KB}) \subseteq I \subseteq \mathit{gfp}(\gamma^2_{KB})$ for every fixpoint $I \subseteq HB_\Phi$ of γ^2_{KB}, it thus follows that $\mathit{lfp}(\gamma^2_{KB}) \subseteq I \subseteq \mathit{gfp}(\gamma^2_{KB})$ for every answer set I of KB. Thus, every such I includes every well-founded and no unfounded atom $a \in HB_\Phi$ relative to KB. □

Proof of Theorem 6.10. If every $a \in HB_\Phi$ is either well-founded or unfounded w.r.t. KB, then $\mathit{lfp}(\gamma^2_{KB}) = \mathit{gfp}(\gamma^2_{KB})$. Thus, $\mathit{lfp}(\gamma^2_{KB}) = I = \mathit{gfp}(\gamma^2_{KB})$, for every fixpoint $I \subseteq HB_\Phi$ of γ_{KB} and so of γ^2_{KB}. That is, $\mathit{lfp}(\gamma^2_{KB}) = I = \mathit{gfp}(\gamma^2_{KB})$ for every answer set I of KB. That is, the set of all well-founded $a \in HB_\Phi$ w.r.t. KB is the only answer set of KB. □

Proof of Theorem 6.11. We use the characterization of $WFS(KB)$ in Theorem 6.7. If KB is positive, then for every $I \subseteq HB_\Phi$, it holds that $P^I = P$, and thus $\gamma_{KB}(I)$ is the least model of KB. Thus, the only fixpoint of γ^2_{KB} (and thus also the least and the greatest fixpoint of γ^2_{KB}) is the least model of KB, which in turn is the unique answer set of KB. □

Appendix C: Proofs for Section 8

Proof of Theorem 8.1. We first prove membership in $\mathrm{NEXP}^{\mathrm{NP}}$. Guessing an interpretation $I \subseteq HB_\Phi$, computing the FLP-reduct P^I of P relative to I, and verifying that I is a model of P^I can be done in nondeterministic exponential time. To verify that I is also a model of L, we then guess an equivalence relation $\sim$ on Φ_c, which can be done in nondeterministic polynomial time, and we verify that (i) $\sim$ is admissible with I,

which can be done in exponential time, and (ii) $L^\star = L \cup (I \cap DL_\Phi) \cup \{\neg b \mid b \in DL_\Phi - I\} \cup \{c \neq c' \mid c \not\sim c'\}$ is satisfiable, which can be done in deterministic (resp., nondeterministic) exponential time, since deciding whether a description logic knowledge base in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$) is satisfiable is in EXP (resp., NEXP), and since $L^\star$ has a polynomial size. In summary, guessing some $I \subseteq HB_\Phi$ and verifying that it is a model of $KB^I = (L, P^I)$ is in NEXP. It then remains to verify that I is also a minimal model of KB^I. This can be done with an exponential-size input to an oracle in NP. Overall, this shows that deciding whether KB has an answer set is in $\mathrm{NEXP}^{\mathrm{NP}}$.

Hardness for $\mathrm{NEXP}^{\mathrm{NP}}$ follows from Theorem 6.3 and the $\mathrm{NEXP}^{\mathrm{NP}}$-hardness of deciding whether an ordinary disjunctive logic program has an answer set [15]. □

Proof of Theorem 8.2. Membership in co-$\mathrm{NEXP}^{\mathrm{NP}}$ (resp., $\mathrm{NEXP}^{\mathrm{NP}}$) follows from the membership in $\mathrm{NEXP}^{\mathrm{NP}}$ of deciding whether a disjunctive dl-program has an answer set (by Theorem 8.1), since a is true in some (resp., every) answer set of KB iff $(L, P \cup \{\leftarrow not\, a\})$ (resp., $(L, P \cup \{\leftarrow a\})$) has an (resp., no) answer set.

Hardness for co-$\mathrm{NEXP}^{\mathrm{NP}}$ (resp., $\mathrm{NEXP}^{\mathrm{NP}}$) follows from the $\mathrm{NEXP}^{\mathrm{NP}}$-hardness of deciding whether a disjunctive dl-program has an answer set (by Theorem 8.1), since KB has an (resp., no) answer set iff p is true in some (resp., every) answer set of $(L, P \cup \{p \leftarrow\})$ (resp., $(L, P \cup \{\leftarrow p\})$), where p is a fresh propositional symbol. □

Proof of Theorem 8.3. Hardness for EXP in both (a) and (b) follows from the EXP-hardness of deciding, given an ordinary positive program P and a ground atom a, whether P logically entails a [15]. Membership in EXP in both (a) and (b) follows from the fact that the fixpoint iterations for computing $WFS(L, P)$ can be done in exponential time, since instance checking and knowledge base satisfiability in *DL-Lite*$_\mathcal{A}$ can be done in polynomial time. □

Appendix D: Proofs for Section 9

Proof of Theorem 9.1. Hardness for P in both (a) and (b) follows from the P-hardness of deciding, given an ordinary positive program P and a ground atom a, whether P logically entails a in the data complexity [15]. Membership in P in both (a) and (b) follows from the fact that the fixpoint iterations for computing $WFS(L, P)$ can be done in polynomial time in the data complexity, since instance checking and knowledge base satisfiability in *DL-Lite*$_\mathcal{A}$ can be done in polynomial time. □

Proof of Theorem 9.2. (a) We first show that deciding whether a given ground literal ℓ belongs to $WFS(KB)$ is first-order rewritable. Since KB is acyclic, there exists a mapping $\kappa\colon \mathcal{P}_P \to \{0, 1, \ldots, n\}$ such that for every rule $r \in P$, the predicate symbol p of $H(r)$, and every predicate symbol q of some $b \in B(r)$, it holds that $\kappa(p) > \kappa(q)$. We call $\kappa(p)$ the *rank* of p. Since every L_i^j is defined in *DL-Lite*$_\mathcal{A}$, as shown in [55], every concept, role, and attribute membership a from L_i^j can be expressed in terms of a first-order formula over the concept, role, and attribute membership axioms in L_i^j. We first show by induction on $\kappa(p) \in \{0, 1, \ldots, n\}$ that every predicate $p \in \mathcal{P}_P$ (relative to $(L \cup P) \setminus L^O$) can also be expressed in terms of a first-order formula over the concept, role, and attribute membership axioms in $L \setminus L^O$ and the *database facts* in P, constructed from predicate symbols of rank 0.

Basis: Every predicate symbol p of rank 0 that does not occur in L, can trivially be expressed in terms of a first-order formula over the database facts in P. As stated above, by [55], every predicate symbol p of rank 0 that occurs in some L_i^j can be expressed in terms of a first-order formula over the concept, role, and

attribute membership axioms in L_i^j. In summary, every predicate symbol p of rank 0 can be expressed in terms of a first-order formula over the concept, role, and attribute membership axioms in $L \setminus L^O$ and the database facts in P.

Induction: Consider any predicate symbol $p \in \mathcal{P}_P$ along with the set of all its defining rules in P, that is, all rules in P with p in their head. W.l.o.g., the heads $p(\mathbf{x})$ of all these rules coincide. Let $\alpha(\mathbf{x})$ denote the disjunction of the existentially quantified bodies of these rules. By the induction hypothesis, every body predicate symbol in $\alpha(\mathbf{x})$ can be expressed in terms of a first-order formula over the concept, role, and attribute membership axioms in L and the database facts in P. Let the first-order formula $\alpha'(\mathbf{x})$ be obtained from $\alpha(\mathbf{x})$ by replacing all atoms by their first-order formulas. Then, $\alpha'(\mathbf{x})$ expresses p in terms of the concept, role, and attribute membership axioms in $L \setminus L^O$ and the database facts in P.

We next add the description logic knowledge base L^O. As stated above, by [55], every predicate symbol p that occurs in L^O can be expressed in terms of a first-order formula $\alpha(\mathbf{x})$ over the concept, role, and attribute membership axioms in L^O. As shown above, every predicate symbol $q \in \mathcal{P}_P$ (relative to $(L \cup P) \setminus L^O$) can be expressed in terms of a first-order formula ϕ over the concept, role, and attribute membership axioms in $L \setminus L^O$ and the database facts in P. Let the first-order formula $\alpha'(\mathbf{x})$ be obtained from $\alpha(\mathbf{x})$ by replacing every atom $q(\mathbf{y})$ by the formula $q(\mathbf{y}) \vee \phi(\mathbf{y})$. Then, $\alpha'(\mathbf{x})$ is a first-order formula for p over the concept, role, and attribute membership axioms in L and the database facts in P.

(b) As for the w-consistency problem, by [55], the satisfiability of L^O and every L_j^I can be expressed in terms of a first-order formula over the concept, role, and attribute membership axioms in L^O and every L_j^I, respectively. The acyclicity of KB implies that we only have to check these satisfiabilities, where all derived facts are added to L^O. The first-order formula for this satisfiability check is constructed as above, and then disjunctively combined with the disjunction of the first-order formulas for all L_j^I. □

References

[1] G. Alsaç and C. Baral. Reasoning in description logics using declarative logic programming. Technical report, Department of Computer Science and Engineering, Arizona State University, 2001.

[2] J. Angele, H. Boley, J. de Bruijn, D. Fensel, P. Hitzler, M. Kifer, R. Krummenacher, H. Lausen, A. Polleres, and R. Studer. Web Rule Language (WRL), Sept. 2005. W3C Member Submission. Available at `http://www.w3.org/Submission/WRL/`.

[3] G. Antoniou. Nonmonotonic rule systems on top of ontology layers. In *Proc. ISWC-2002*, *LNCS* 2342, pp. 394–398. Springer, 2002.

[4] G. Antoniou, C. V. Damásio, B. Grosof, I. Horrocks, M. Kifer, J. Maluszynski, and P. F. Patel-Schneider. Combining rules and ontologies: A survey. Technical Report IST506779/Linköping/I3-D3/D/PU/a1, Linköping University, February 2005.

[5] J. Bao, E. F. Kendall, D. L. McGuinness, and E. K. Wallace. OWL 2 Web ontology language: Quick reference guide, 2008. Available at `http://www.w3.org/TR/owl2-quick-reference/`.

[6] C. Baral and V. S. Subrahmanian. Dualities between alternative semantics for logic programming and nonmonotonic reasoning. *J. Autom. Reasoning*, 10(3):399–420, 1993.

[7] T. Berners-Lee. *Weaving the Web*. Harper, San Francisco, CA, 1999.

[8] A. Calì, G. Gottlob, and T. Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. In *Proc. PODS-2009*, pp. 77–86. ACM Press, 2009.

[9] A. Calì, G. Gottlob, and T. Lukasiewicz. Datalog$^{\pm}$: A unified approach to ontologies and integrity constraints. In *Proc. ICDT-2009*, pp. 14–30. ACM Press, 2009.

[10] A. Calì and T. Lukasiewicz. Tightly integrated probabilistic description logic programs for the Semantic Web. In *Proc. ICLP-2007*, *LNCS* 4670, pp. 428–429. Springer, 2007.

[11] A. Calì and T. Lukasiewicz. An approach to probabilistic data integration for the Semantic Web. In *Uncertainty Reasoning for the Semantic Web I*, *LNCS* 5327, pp. 52–65. Springer, 2008.

[12] A. Calì, T. Lukasiewicz, L. Predoiu, and H. Stuckenschmidt. Tightly integrated probabilistic description logic programs for representing ontology mappings. In *Proc. FoIKS-2008*, *LNCS* 4932, pp. 178–198. Springer, 2008.

[13] A. Calì, T. Lukasiewicz, L. Predoiu, and H. Stuckenschmidt. Tightly coupled probabilistic description logic programs for the Semantic Web. *Journal on Data Semantics*, 12:95–130, 2009.

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

[15] E. Dantsin, T. Eiter, G. Gottlob, and A. Voronkov. Complexity and expressive power of logic programming. *ACM Comput. Surv.*, 33(3):374–425, 2001.

[16] J. de Bruijn, T. Eiter, A. Polleres, and H. Tompits. Embedding non-ground logic programs into autoepistemic logic for knowledge base combination. In *Proc. IJCAI-2007*, pp. 304–309. AAAI Press/IJCAI, 2007.

[17] J. de Bruijn, H. Lausen, A. Polleres, and D. Fensel. The Web service modeling language WSML: An overview. In *Proc. ESWC-2006*, *LNCS* 4011, pp. 590–604. Springer, 2006.

[18] F. M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. $\mathcal{AL}$-log: Integrating Datalog and description logics. *J. Intell. Inf. Syst.*, 10(3):227–252, 1998.

[19] W. Drabent, T. Eiter, G. Ianni, T. Krennwallner, T. Lukasiewicz, and J. Małuszyński. Hybrid reasoning with rules and ontologies. In F. Bry and J. Małuszyński, editors, *Semantic Techniques for the Web: The REWERSE Perspective*, *LNCS* 5500, pp. 1–49. Springer, 2009.

[20] W. Drabent and J. Małuszyński. Well-founded semantics for hybrid rules. In *Proc. RR-2007*, *LNCS* 4524, pp. 1–15. Springer, 2007.

[21] T. Eiter, G. Ianni, T. Lukasiewicz, R. Schindlauer, and H. Tompits. Combining answer set programming with description logics for the Semantic Web. *Artif. Intell.*, 172(12/13):1495–1539, 2008.

[22] T. Eiter, G. Ianni, A. Polleres, R. Schindlauer, and H. Tompits. Reasoning with rules and ontologies. In P. Barahona, F. Bry, E. Franconi, N. Henze, and U. Sattler, editors, *Reasoning Web, Second International Summer School 2006, Tutorial Lectures*, *LNCS* 4126, pp. 93–127. Springer, 2006.

[23] T. Eiter, G. Ianni, R. Schindlauer, and H. Tompits. A uniform integration of higher-order reasoning and external evaluations in answer-set programming. In *Proc. IJCAI-2005*, pp. 90–96. Professional Book Center, 2005.

[24] T. Eiter, G. Ianni, R. Schindlauer, and H. Tompits. Effective integration of declarative rules with external evaluations for Semantic Web reasoning. In *Proc. ESWC-2006*, *LNCS* 4011, pp. 273–287. Springer, 2006.

[25] T. Eiter, T. Lukasiewicz, R. Schindlauer, and H. Tompits. Combining answer set programming with description logics for the Semantic Web. In *Proc. KR-2004*, pp. 141–151. AAAI Press, 2004.

[26] T. Eiter, T. Lukasiewicz, R. Schindlauer, and H. Tompits. Well-founded semantics for description logic programs in the Semantic Web. In *Proc. RuleML-2004*, *LNCS* 3323, pp. 81–97. Springer, 2004.

[27] W. Faber, N. Leone, and G. Pfeifer. Recursive aggregates in disjunctive logic programs: Semantics and complexity. In *Proc. JELIA-2004*, *LNCS* 3229, pp. 200–212. Springer, 2004.

[28] D. Fensel, W. Wahlster, H. Lieberman, and J. Hendler, editors. *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press, 2002.

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

[30] B. N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs: Combining logic programs with description logics. In *Proc. WWW-2003*, pp. 48–57. ACM Press, 2003.

[31] S. Heymans, D. V. Nieuwenborgh, and D. Vermeir. Nonmonotonic ontological and rule-based reasoning with extended conceptual logic programs. In *Proc. ESWC-2005*, *LNCS* 3532, pp. 392–407. Springer, 2005.

[32] S. Heymans and D. Vermeir. Integrating Semantic Web reasoning and answer set programming. In *Proc. ASP-2003*, *CEUR Workshop Proceedings* 78, pp. 194–208. CEUR-WS.org, 2003.

[33] I. Horrocks and P. F. Patel-Schneider. Reducing OWL entailment to description logic satisfiability. In *Proc. ISWC-2003*, *LNCS* 2870, pp. 17–29. Springer, 2003.

[34] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A Semantic Web rule language combining OWL and RuleML, May 2004. W3C Member Submission. Available at `http://www.w3.org/Submission/SWRL/`.

[35] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *J. Web Sem.*, 1(1):7–26, 2003.

[36] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In *Proc. LPAR-1999*, *LNCS* 1705, pp. 161–180. Springer, 1999.

[37] U. Hustadt, B. Motik, and U. Sattler. Reducing $\mathcal{SHIQ}$-description logic to disjunctive Datalog programs. In *Proc. KR-2004*, pp. 152–162. AAAI Press, 2004.

[38] D. S. Johnson. A catalog of complexity classes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A, chapter 2, pp. 67–161. MIT Press, Cambridge, MA, 1990.

[39] M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *J. ACM*, 42(4):741–843, 1995.

[40] M. Knorr, J. J. Alferes, and P. Hitzler. A coherent well-founded model for hybrid MKNF knowledge bases. In *Proc. ECAI-2008*, *Frontiers in Artificial Intelligence and Applications* 178, pp. 99–103. IOS Press, 2008.

[41] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello. The DLV system for knowledge representation and reasoning. *ACM Trans. Comput. Log.*, 7(3):499–562, 2006.

[42] A. Y. Levy and M.-C. Rousset. Combining Horn rules and description logics in CARIN. *Artif. Intell.*, 104(1–2):165–209, 1998.

[43] T. Lukasiewicz. Fuzzy description logic programs under the answer set semantics for the Semantic Web. In *Proc. RuleML-2006*, pp. 89–96. IEEE Computer Society, 2006.

[44] T. Lukasiewicz. Fuzzy description logic programs under the answer set semantics for the Semantic Web. *Fundam. Inform.*, 82(3):289–310, 2008.

[45] T. Lukasiewicz. Probabilistic description logic programs. In *Proc. ECSQARU-2005*, *LNCS* 3571, pp. 737–749. Springer, 2005.

[46] T. Lukasiewicz. Probabilistic description logic programs. *Int. J. Approx. Reasoning*, 45(2):288–307, 2007.

[47] T. Lukasiewicz. Tractable probabilistic description logic programs. In *Proc. SUM-2007*, *LNCS* 4772, pp. 143–156. Springer, 2007.

[48] T. Lukasiewicz and U. Straccia. Tightly integrated fuzzy description logic programs under the answer set semantics for the Semantic Web. In *Proc. RR-2007*, *LNCS* 4524, pp. 289–298. Springer, 2007.

[49] T. Lukasiewicz and U. Straccia. Tightly coupled fuzzy description logic programs under the answer set semantics for the Semantic Web. *Int. J. Semantic Web Inf. Syst.*, 4(3):68–89, 2008.

[50] B. Motik, I. Horrocks, R. Rosati, and U. Sattler. Can OWL and logic programming live together happily ever after? In *Proc. ISWC-2006*, *LNCS* 4273, pp. 501–514. Springer, 2006.

[51] B. Motik and R. Rosati. A faithful integration of description logics with logic programming. In *Proc. IJCAI-2007*, pp. 477–482. AAAI Press/IJCAI, 2007.

[52] B. Motik, U. Sattler, and R. Studer. Query answering for OWL-DL with rules. *J. Web Sem.*, 3(1):41–60, 2005.

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

[54] P. F. Patel-Schneider and I. Horrocks. Position paper: A comparison of two modelling paradigms in the Semantic Web. In *Proc. WWW-2006*, pp. 3–12. ACM Press, 2006.

[55] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *J. Data Semantics*, 10:133–173, 2008.

[56] R. Rosati. On the decidability and complexity of integrating ontologies and rules. *J. Web Sem.*, 3(1):61–73, 2005.

[57] R. Rosati. Integrating ontologies and rules: Semantic and computational issues. In P. Barahona, F. Bry, E. Franconi, N. Henze, and U. Sattler, editors, *Reasoning Web*, *LNCS* 4126, pp. 128–151. Springer, 2006.

[58] R. Rosati. $\mathcal{DL}+log$: Tight integration of description logics and disjunctive Datalog. In *Proc. KR-2006*, pp. 68–78. AAAI Press, 2006.

[59] M. Sintek and S. Decker. TRIPLE - A query, inference, and transformation language for the Semantic Web. In *Proc. ISWC-2002*, *LNCS* 2342, pp. 364–378. Springer, 2002.

[60] T. Swift. Deduction in ontologies via ASP. In *Proc. LPNMR-2004*, *LNCS* 2923, pp. 275–288. Springer, 2004.

[61] S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. PhD thesis, RWTH Aachen, Germany, 2001.

[62] A. van Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *J. ACM*, 38(3):620–650, 1991.

[63] W3C. OWL web ontology language overview, 2004. W3C Recommendation (10 Feb. 2004). Available at `http://www.w3.org/TR/2004/REC-owl-features-20040210/`.