Eliminating Nonmonotonic DL-atoms in Description Logic Programs

Yisong Wang[1], Thomas Eiter[2], Jia-Huai You[3], LiYan Yuan[3], and Yi-Dong Shen[4]

[1] Department of Computer Science, Guizhou University, Guiyang, China
[2] Institute of Information Systems, Vienna University of Technology, Austria
[3] Department of Computing Science, University of Alberta, Canada
[4] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China

Abstract. Nonmonotonic description logic programs (dl-programs) are a well-known formalism for combining rules and ontologies, where rules interact with an underlying ontology via dl-atoms that allow queries to the ontology under a possible update of its assertional part. It is known that dl-atoms may be nonmonotonic and dl-programs without nonmonotonic dl-atoms have many desirable properties. In this paper, we show that it is possible to remove nonmonotonic dl-atoms from a dl-program while preserving its strong/weak answer set semantics. Though the translation is faithful, it relies on the knowledge about monotonicity of dl-atoms. We then thoroughly investigate the complexity of deciding whether a dl-atom is monotonic under the description logics DL-Lite$_\mathcal{R}$, $\mathcal{EL}^{++}$, $\mathcal{SHIF}$ and $\mathcal{SHOIN}$, which is of independent interest for computing strong answer sets. We show that the problem is intractable in general, but tractable for dl-atoms with bounded queries in DL-Lite$_\mathcal{R}$.

1 Introduction

Logic programming under the answer set semantics (ASP) has been recognized as an expressive nonmonotonic reasoning framework for declarative knowledge representation [4]. Recently, there has been an extensive interest in combining ASP with description logics (DLs) for the Semantic Web [7,8,11,16,18,21,22], see [6,18] for a comprehensive overview. While most approaches are based on model building either within a nonmonotonic modal logic like MKNF [18] and AEL [7], or over hybrid formulas including QEL [8,16], the approach of dl-programs [11] stands out by facilitating inference-based communication via well-designed interfaces called *dl-atoms*. This enables one to reason on top of queries to ontologies, where the rules can combine the query results in a subtle manner. The approach is closely related to equipping logic programs with external sources [12] and has been implemented e.g. in dlvhex and DReW.[5]

Informally, a dl-program is a pair (O, P), where O is an ontology (i.e., a knowledge base) expressed in a description logic, and P a logic program, where rule bodies may contain queries to the knowledge base O, called *dl-atoms*. Such queries allow to specify inputs from a logic program to the knowledge base. In more detail, a dl-atom is of the

[5] See `www.kr.tuwien.ac.at/research/systems/` link `dlvhex/` resp. `drew/`

form $DL[\lambda; Q](\boldsymbol{t})$, where $Q(\boldsymbol{t})$ is a query to O, and λ is a list $S_1\ op_1\ p_1, \ldots, S_m\ op_m\ p_m$ of "virtual updates" $S_i\ op_i\ p_i$, where S_i is a concept or a role in O, p_i is a predicate symbol in P of the same arity, and op_i is one of the operators $\uplus, \cup, \cap$. Intuitively, $\uplus$ (resp., $\cup$) increases S_i (resp., $\neg S_i$) by the extension of p_i, while $\cap$ (called the *constraint operator*) restricts S_i to p_i by closed world assumption, i.e., in the absence of $p_i(\boldsymbol{c})$, we assert $\neg S_i(\boldsymbol{c})$; notably, $\uplus$ and $\cup$ are monotonic in p_i, while $\cap$ is anti-monotonic. Dl-programs under weak and strong answer sets semantics [11] were further investigated from the perspectives of loop formulas [25] and the logic of here-and-there [14].

A dl-atom may be nonmonotonic, due to the presence of the constraint operator. Dl-programs without nonmonotonic dl-atoms, which we call *canonical*, enjoy many desirable properties. For example, the strong answer sets of a canonical dl-program are minimal under set inclusion; if in addition no default negation occurs in it (i.e., the dl-program is *positive*), then it has a unique answer set that is given but the least fixpoint of an immediate consequence operator. In addition, we know that a key component in most state-of-the-art ASP solvers is the constraint propagation in terms of computing a well-founded model that extends the current partial assignment. For canonical dl-programs, as shown in [13], the well-founded model can be computed by a quadratic number of calls to the underlying ontology, in the size of a ground dl-program. Thus, if query-answering in an ontology is tractable, constraint propagation in terms of computing the well-founded model is also tractable. Solvers for dl-programs can take advantage of this, as the well-founded model approximates the strong answer sets; furthermore, characterizations of strong answer sets (e.g., in terms of unfounded sets, cf. [9]) can be exploited. Therefore, eliminating nonmonotonic dl-atoms from a dl-program is of practical relevance in answer set computation.

In this paper, we show that any dl-program can be faithfully transformed into a canonical dl-program under the strong/weak answer set semantics. This result is not only of use for computational purposes as mentioned above, but also allows to extend embeddings for the class of canonical dl-programs under strong answer set semantics into MKNF [18] and First-Order Autoepistemic Logic (FO-AEL) [7] to all dl-programs.

However, one aspect of our translation is that it relies on the knowledge whether dl-atoms occurring in dl-programs are monotonic. We thus investigate the complexity of deciding this property, which is of independent interest, as in [11] knowledge about monotonic dl-atoms may be used as a parameter to define strong answer sets. We concentrate in this paper on the description logics $\mathcal{SHIF}$ and $\mathcal{SHOIN}$ considered in [11], and on the tractable description logics DL-Lite$_\mathcal{R}$ [5] and $\mathcal{EL}^{++}$ [2].

While absence of the constraint operator from a dl-atom implies monotonicity, this is not a necessary condition. In fact, the dl-atom $A = DL[S' \uplus p, S' \cup p, S \cap p; \neg S](a)$ evaluates always to true, regardless of the underlying ontology, and thus is monotonic; such tautological (and analogous contradictory) dl-atoms can be efficiently recognized [10]. However, in general recognizing monotonic dl-atoms turns out to be intractable, even for tractable classes of ontologies including DL-Lite$_\mathcal{R}$ and $\mathcal{EL}^{++}$. On the other hand, we show that if the size of the query in the dl-atom is bounded, the problem becomes tractable for DL-Lite$_\mathcal{R}$. Thus in this setting—which is important in practice—we obtain a faithful and polynomial translation to eliminate nonmonotonic dl-atoms from arbitrary dl-programs.

2 Preliminaries

In this section, we briefly review basic notions of description logics [3] in an abstract manner and description logic programs [11].

2.1 Description logics

The vocabulary of a description logic (DL) includes disjoint sets of *individual names* $\mathbf{I}$, *concept names* $\mathbf{C}$ (also called *atomic concepts*), and *role names* $\mathbf{R}$ (also called *atomic roles*). Intuitively, individuals names are constants, concept names are unary predicate symbols, and role names are binary predicate symbols. Complex concept and role expressions C and R, respectively, in core DLs are formed using Boolean connectives ($\neg$ for negation, $\sqcap$ for conjunction and $\sqcup$ for disjunction) and existential restriction, as well as predicate inversion, under syntactic restrictions.

A description logic knowledge base (or ontology) is a pair $O = \langle \mathcal{T}, \mathcal{A} \rangle$ where
— $\mathcal{T}$, the *TBox*, is a finite set of formulas $X \sqsubseteq Y$, where X and Y are (restricted) concept (resp. role) expressions, called *concept (resp. role) inclusion axioms*;

— $\mathcal{A}$, the *ABox*, is a finite set of formulas $X(\boldsymbol{a})$, where X is a (restricted) concept or role expression and $\boldsymbol{a}$ is a tuple of individual names matching the arity of X, called *membership assertions*.

While $\mathcal{T}$ specifies terminological knowledge, $\mathcal{A}$ specifies extensional knowledge. The semantics of O is natively defined in terms of models similarly as for theories in predicate logic, or alternatively for many DLs by a translation $\tau(O)$ of O into first-order logic [3]. An ontology O is *satisfiable* whenever $\tau(O)$ has a model. A *dl-query* q is either an inclusion axiom or a membership assertion. By $O \models q$ we denote that every model of $\tau(O)$ is a model of $\tau(q)$. In general, deciding $O \models q$ is reducible to deciding satisfiability of an ontology O'. In what follows, we assume that the underlying ontologies are from a class for which the satisfiability problem is decidable (which usually is the case). There are many well-known such classes, e.g. DL-Lite$_{\mathcal{R}}$ [5] and $\mathcal{EL}^{++}$ [2], for which it is tractable, and $\mathcal{SHIF}$ and $\mathcal{SHOIN}$ [15], for which it is EXP-complete resp. NEXP-complete [19].

2.2 Description logic programs: syntax and models

Let $\Phi = (\mathcal{P}, \mathcal{C})$ be a first-order vocabulary with finite nonempty sets $\mathcal{P}$ of predicate symbols and $\mathcal{C} \subseteq \mathbf{I}$ of constant symbols such that $\mathcal{P}$ is disjoint from $\mathbf{C} \cup \mathbf{R}$. *Ordinary atoms* (simply *atoms*) are formed from $\mathcal{P}$, $\mathcal{C}$, and variables as usual.

Definition 1 (Dl-atoms). *A* dl-atom *is an expression of the form*

$$DL[\lambda; Q](\boldsymbol{t}), \quad (m \geq 0) \tag{1}$$

where $\lambda = S_1\ op_1\ p_1, \ldots, S_m\ op_m\ p_m$ *and for all* i $(1 \leq i \leq n)$,
- $op_i \in \{\uplus, \cup, \cap\}$ *(we call* $\cap$ *the* constraint operator*);*
- S_i *is a concept (resp. role) whenever* p_i *is a unary (resp. binary) predicate;*
- $Q(\boldsymbol{t})$ *is a dl-query scheme, i.e. an inclusion axiom or a membership assertion where* $\boldsymbol{t}$ *may contain variables as placeholders for constants.*

The p_i are called *input predicates*. Intuitively, $S_i \uplus p_i$ extends S_i by the extension of p_i, and $S_i \uplus p_i$ analogously extends $\neg S_i$; the expression $S \cap p_i$ instead constrains S_i to p_i. E.g., $DL[S \uplus p; S](a)$ is a dl-atom in which the concept S is extended with the extension of p prior to querying whether a is an instance of S; the dl-atom $DL[S \uplus p, R \cap q; S](a)$ is similar but in addition the role R is constrained to the extension of q.

A *dl-rule* (or simply a *rule*) r is an expression of the form

$$A \leftarrow B_1, \ldots, B_m, not\ B_{m+1}, \ldots, not\ B_n, (n \geq m \geq 0) \tag{2}$$

where A is an atom and each B_i $(1 \leq i \leq n)$ is either an ordinary atom or a dl-atom.[6] We refer to A as the *head* of r and to the conjunction of all B_i $(1 \leq i \leq m)$ and $not\ B_j$ $(m+1 \leq j \leq n)$ as the *body* of r. For convenience, we view r also of the form

$$A \leftarrow \mathit{Pos}, not\ \mathit{Neg} \tag{3}$$

where $\mathit{Pos} = \{B_1, \ldots, B_m\}$, $\mathit{Neg} = \{B_{m+1}, \ldots, B_n\}$ and $not\ S = \{not\ A \mid A \in S\}$ for a set S of atoms or dl-atoms. A *ground instance* of a dl-rule r is obtained from r by replacing every variable occurring in r, by a constant symbol from $\mathcal{C}$.

A *description logic (dl-)program* $\mathcal{K}$ is a pair (O, P) where O is an ontology and P is a finite set of dl-rules. By $\mathit{ground}(P)$ we denote the set of all ground instances of the rules in P. In what follows, we assume that the vocabulary $\mathcal{P}$ and $\mathcal{C}$ of P is implicitly given by the predicate and constant symbols occurring in P respectively and that P is *ground* (viz. $P = \mathit{ground}(P)$) unless explicitly stated otherwise.

Given a dl-program $\mathcal{K} = (O, P)$, the *Herbrand base* of P, denoted by HB_P, is the set of all ground atoms $p(c_1, \ldots, c_n)$ with predicate symbol $p \in \mathcal{P}$ and arguments $c_i \in \mathcal{C}$. A (Herbrand) *interpretation* (relative to P) is a subset $I \subseteq \mathit{HB}_P$; the atoms in I (resp., $\mathit{HB}_P \setminus I$) are assigned being true (resp., false). Given $P = \mathit{ground}(P)$, we can restrict HB_P for our purposes to the ordinary atoms occurring in P and all ground atoms $p(\boldsymbol{c})$ where p occurs in some dl-atom of P. Under this restriction, the size of HB_P is polynomial in the size of P.

An interpretation I *satisfies* (is a *model* of) an (ordinary or dl-)atom A under O, denoted $I \models_O A$, if the following holds:

- if $A \in \mathit{HB}_P$, then $I \models_O A$ iff $A \in I$;
- if $A = DL[S_1\ op_1\ p_1, \ldots, S_m\ op_m\ p_m; Q](\boldsymbol{t})$, then $I \models_O A$ iff $O(I; \lambda) \models Q(\boldsymbol{t})$ [7] where $O(I; \lambda) = O \cup A(I)$, $A(I) = \bigcup_{i=1}^{m} A_i(I)$ and, for $1 \leq i \leq m$,

$$A_i(I) = \begin{cases} \{S_i(\boldsymbol{c}) \mid p_i(\boldsymbol{c}) \in I\}, & \text{if } op_i = \uplus; \\ \{\neg S_i(\boldsymbol{c}) \mid p_i(\boldsymbol{c}) \in I\}, & \text{if } op_i = \uplus; \\ \{\neg S_i(\boldsymbol{c}) \mid p_i(\boldsymbol{c}) \in \mathit{HB}_P \setminus I\}, & \text{if } op_i = \cap, \end{cases}$$

An interpretation I is a *model* of a dl-rule of the form (3), if $I \models_O B$ for each $B \in \mathit{Pos}$ and $I \not\models_O B'$ for each $B' \in \mathit{Neg}$ implies $I \models_O A$. An interpretation I is a *model* of a dl-program $\mathcal{K} = (O, P)$, denoted $I \models_O \mathcal{K}$, if I is a model of each rule of P.

[6] Unlike [11], we omit strong negation here for simplicity and without loss of expressiveness.

[7] Technically, one assumes that for $O(I; \lambda)$ beyond the syntax of an ontology class, $O(I; \lambda) \models Q(\boldsymbol{t})$ can be recast to basic reasoning (e.g. satisfiability) on an ontology from the same class.

We note that different forms of dl-atoms might be equivalent; e.g., $DL[A \uplus p; Q](\boldsymbol{t})$ and $DL[\neg A \uplus p; Q](\boldsymbol{t})$ have this property, and the same holds for $DL[A \uplus p, B \uplus p; Q](\boldsymbol{t})$ and $DL[A \sqcap B \uplus p; Q](\boldsymbol{t})$. In fact, the following statement holds.

Lemma 1. *The dl-atoms $DL[\lambda; Q](\boldsymbol{t})$ and $DL[\lambda'; Q](\boldsymbol{t})$ have same models, if λ' results from λ by replacing "$S_1 \uplus p, \ldots, S_k \uplus p$" with $(S_1 \sqcap \cdots \sqcap S_k) \uplus p$, "$(S_1 \sqcup \cdots \sqcup S_n) \uplus p$" with $(\neg S_1 \sqcap \cdots \sqcap \neg S_n) \uplus p$, or "$S_1 \uplus p, \ldots, S_n \uplus p$" with $(S_1 \sqcup \cdots \sqcup S_k) \uplus p$,.*

Monotonicity. A dl-atom A is *monotonic* (relative to a dl-program $\mathcal{K} = (O, P)$), if $I \models_O A$ implies $I' \models_O A$, for all I' such that $I \subseteq I' \subseteq HB_P$; otherwise A is *nonmonotonic*. Clearly, a dl-atom A is monotonic if the constraint operator $\cap\!\!\!-$ does not occur in A; however, A can be monotonic even if $\cap\!\!\!-$ occurs.

Example 1. The dl-atom $A = DL[S' \uplus p, S' \uplus p, S \cap\!\!\!- p; \neg S](a)$ from the Introduction is tautological, i.e., $I \models_O A$ for every interpretation $I \subseteq HB_P$ of any given dl-program $\mathcal{K} = (O, P)$. On the other hand, $A = DL[S \cap\!\!\!- p; C](a)$ is not tautological, but clearly monotonic w.r.t. $\mathcal{K} = (O, P)$ if $O = \emptyset$.

Given a dl-program $\mathcal{K} = (O, P)$, we denote by DL_P the set of all dl-atoms that occur in P, by $DL_P^+ \subseteq DL_P$ the set of all monotonic dl-atoms, and by $DL_P^- = DL_P \setminus DL_P^+$ the set of all nonmonotonic dl-atoms. Note that in [11], DL_P^+ is any subset of the monotonic dl-atoms (intuitively, those *known* to be monotonic), and all others are regarded as nonmonotonic; we adopt here the ideal (and preferable) case of *complete knowledge* about monotonicity, which can be established whenever the underlying description logic is decidable. A dl-program $\mathcal{K} = (O, P)$ is

- *positive*, if (i) P is "not"-free, and (ii) each dl-atom in P is monotonic relative to $\mathcal{K}$.
- *canonical*, if $DL_P^- = \emptyset$ (i.e., P contains only monotonic dl-atoms);
- *normal*, the constraint operator $\cap\!\!\!-$ does not occur in monotonic dl-atoms of P.

Clearly, every positive $\mathcal{K}$ is canonical, and $\mathcal{K}$ is canonical if $\cap\!\!\!-$ does not occur in P. Normal programs are incomparable to both positive and canonical programs,

Example 2. Let $\mathcal{K} = (\emptyset, P)$ where P consists of

$$p(a) \leftarrow DL[S \uplus p; S](a), \qquad p(a) \leftarrow not\ DL[S \uplus p; S](a).$$

This dl-program is canonical and normal, but not positive. Intuitively, P expresses reasoning by cases: regardless of whether the dl-atom $A = DL[S \uplus p; S](a)$ evaluates to true or false, $p(a)$ should be true.

Positive dl-programs have attractive semantic properties, e.g., they have a least model (under set inclusion). Note that monotonicity of a dl-atom A in a normal dl-program $\mathcal{K}$ is easily decided, as it amounts to the occurrence of $\cap\!\!\!-$ in A.

2.3 Strong and weak answer sets

Let $\mathcal{K} = (O, P)$ be a positive dl-program. The immediate consequence operator $\gamma_\mathcal{K} : 2^{HB_P} \rightarrow 2^{HB_P}$ is defined as, for each $I \subseteq HB_P$,

$$\gamma_\mathcal{K}(I) = \{h \mid h \leftarrow Pos \in P \text{ and } I \models_O A \text{ for each } A \in Pos\},$$

Since $\gamma_\mathcal{K}$ is monotone, it has a least fixpoint $lfp(\gamma_\mathcal{K})$ which coincides with the least model of $\mathcal{K}$ and can be iteratively constructed as $\gamma_\mathcal{K}^\infty = \bigcup_{n \geq 0} \gamma_\mathcal{K}^n$ where

$$\gamma_{\mathcal{K}}^0 = \emptyset, \text{ and } \gamma_{\mathcal{K}}^{n+1} = \gamma_{\mathcal{K}}(\gamma_{\mathcal{K}}^n) \text{ for } n \geq 0.$$

Let $\mathcal{K} = (O, P)$ be a dl-program. The *strong dl-transform* of $\mathcal{K}$ relative to O and an interpretation $I \subseteq HB_P$, denoted by $\mathcal{K}^{s,I}$, is the positive dl-program (O, sP_O^I), where sP_O^I is obtained from P by deleting:

- every dl-rule r of the form (2) such that either $I \not\models_O B_i$ for some $i \in \{1, \ldots, m\}$ and $B_i \in DL_P^-$, or $I \models_O B_j$ for some $j \in \{m+1, \ldots, n\}$;[8] and
- all nonmonotonic dl-atoms and $not\ A$ from the remaining dl-rules where A is an atom or a dl-atom.

The interpretation I is a *strong answer set* of $\mathcal{K}$ if it is the least model of $\mathcal{K}^{s,I}$, i.e., $I = lfp(\gamma_{\mathcal{K}^{s,I}})$.

The *weak dl-transform* of $\mathcal{K}$ relative to O and an interpretation $I \subseteq HB_P$, denoted by $\mathcal{K}^{w,I}$, is the positive dl-program (O, wP_O^I), where wP_O^I is obtained similarly as sP_O^I but with DL_P in place of DL_P^-. An interpretation I is a *weak answer set* of $\mathcal{K}$, if I is the least model of $\mathcal{K}^{w,I}$, i.e., $I = lfp(\gamma_{\mathcal{K}^{w,I}})$. If in a dl-program $\mathcal{K} = (O, P)$ no dl-atom occurs positively, then its weak and strong answer sets coincide (as $\mathcal{K}^{w,I} = \mathcal{K}^{s,I}$ for any interpretation I). The same holds if O is unsatisfiable; in this case, the strong and weak answer sets are incomparable w.r.t. inclusion. This, however, is not true in general.

It has been shown that if a dl-program contains only monotonic dl-atoms, then its strong answer sets are minimal models (cf. Theorem 4.13 of [11]), and thus incomparable w.r.t set inclusion. However, this does not hold for weak answer sets, even if a dl-program is positive. It is also known that strong answer sets are always weak answer sets, but not vice versa. One might wonder whether for each weak answer set I of a dl-program $\mathcal{K}$, it has some strong answer set I' such that $I' \subseteq I$. As illustrated by the following example, this is not the case.

Example 3 (cont'd). Reconsider $\mathcal{K}$ in Example 2, and let $I = \{p(a)\}$. We have that $wP_O^I = \{\, p(a) \,\}$, thus I is a weak answer set of $\mathcal{K}$. However, note that $sP_O^I = \{p(a) \leftarrow DL[S \uplus p; S](a)\}$. The least model of $\mathcal{K}^{s,I}$ is $\emptyset$ $(\neq I)$. So that I is not a strong answer set of $\mathcal{K}$. Now consider $I' = \emptyset$. We have $sP_O^{I'} = \{p(a) \leftarrow DL[S \uplus p; S](a);\ \ p(a)\,\}$. The least model of $\mathcal{K}^{s,I'}$ is $\{p(a)\}$ $(\neq I')$. Thus I' is not a strong answer set of $\mathcal{K}$. In fact, $\mathcal{K}$ has no strong answer sets at all. This is in line with the intuition that, as $O = \emptyset$, $p(a)$ can not be foundedly derived without the assumption that $p(a)$ is true.

3 Eliminating Constraint Operators from Nonmonotonic Dl-atoms

Intuitively, translating a nonmonotonic dl-atom into a monotonic one is to replace $S \cap p$ with $S \cup p'$, where p' is a fresh predicate of the same arity as p that stands for the negation of p. In what follows, we show that the constraint operator can be eliminated from nonmonotonic dl-atoms while preserving strong answer sets. As mentioned previously, we assume that the signatures $\mathcal{P}$ and $\mathcal{C}$ are implicitly given by a dl-program $\mathcal{K}$; any predicate symbol not occurring in $\mathcal{K}$ is fresh.

Definition 2 ($\pi(\mathcal{K})$). *Let $\mathcal{K} = (O, P)$ be a dl-program. Then $\pi(\mathcal{K}) = (O, \pi(P))$, where $\pi(P) = \bigcup_{r \in P} \pi(r)$ and $\pi(r)$ consists for a rule r of form (2) of:*

[8] [11] used $DL_P^?$ in place of DL_P^-, which in our ideal setting of complete knowledge coincides with $DL_P^?$; technically, the discrepancy is not essential.

(i) the rule

$$A \leftarrow \pi(B_1), \ldots, \pi(B_m), \pi(not\, B_{m+1}), \ldots, \pi(not\, B_n),\ where \tag{4}$$

$$\pi(B) = \begin{cases} B, & \text{if } B \text{ is an atom or a monotonic dl-atom;} \\ not\, \pi_B, & \text{if } B \text{ is a nonmonotonic dl-atom,} \end{cases}$$

where π_B is a fresh propositional atom, and

$$\pi(not\, B) = \begin{cases} not\, B, & \text{if } B \text{ is an atom;} \\ not\, \boldsymbol{DL}[\pi(\lambda); Q](\boldsymbol{t}), & \text{if } B = \boldsymbol{DL}[\lambda; Q](\boldsymbol{t}), \end{cases}$$

where $\pi(\lambda)$ results from λ by replacing each "$S \uplus p$" with "$S \uplus \pi_p$", where π_p is a fresh predicate of the same arity as p;

(ii) for each nonmonotonic dl-atom $B \in \{B_1, \ldots, B_m\}$, the rule

$$\pi_B \leftarrow \pi(not\, B) \tag{5}$$

where π_B is from (i) and

(iii) for each predicate p such that "$S \uplus p$" occurs in some nonmonotonic dl-atom of r, all ground instances of the rule

$$\pi_p(\boldsymbol{x}) \leftarrow not\, p(\boldsymbol{x}) \tag{6}$$

where $\boldsymbol{x}$ is a tuple of distinct variables matching the arity of p, and π_p is from (i).

Intuitively, the idea behind π is the following. Recall that "$S \uplus p$" means "infer $\neg S(\boldsymbol{c})$ in the absence of $p(\boldsymbol{c})$". Thus if $\pi_p(\boldsymbol{c})$ stands for the absence of $p(\boldsymbol{c})$ then "$S \uplus p$" should have the same meaning as that of "$S \uplus \pi_p$". Thus, a nonmonotonic dl-atom is expressed by a monotonic dl-atom and nonmonotonic negation "*not*". Note that $\pi(P)$ may still contain dl-atoms with the constraint operator, but they are all monotonic, i.e., $\pi(\mathcal{K})$ is canonical but not necessarily normal. The trick for eliminating unnegated nonmonotonic dl-atoms is simulating "double negation" (allowed in nested expressions [17]), which also is exploited to eliminate the constraint operator from nonmonotonic dl-atoms.

Example 4. Let $\mathcal{K}_1 = (\emptyset, P_1)$, where $P_1 = \{p(a) \leftarrow not\, \boldsymbol{DL}[\neg S \uplus p; S](a)\}$; note that $\mathcal{K}_1$ is normal but neither canonical nor positive. One can check that $\mathcal{K}_1$ has the weak answer sets $\emptyset$ and $\{p(a)\}$, which are both also strong answer sets. Under the translation π, we have $\pi(\mathcal{K}_1) = (\emptyset, \pi(P_1))$, where $\pi(P_1)$ consists of the rules

$$p(a) \leftarrow not\, \boldsymbol{DL}[\neg S \uplus \pi_p; S](a), \qquad \pi_p(a) \leftarrow not\, p(a),$$

which has the weak answer sets $\{p(a)\}$ and $\{\pi_p(a)\}$ corresponding to $\{p(a)\}$ and $\emptyset$, respectively, when restricted to $\boldsymbol{HB}_{P_1}$; again, both are also strong answer sets.

Given a dl-program $\mathcal{K} = (O, P)$ and $I \subseteq \boldsymbol{HB}_P$, let $\pi(I) = I \cup \pi_1(I) \cup \pi_2(I)$ where

$$\pi_1(I) = \{\pi_p(\boldsymbol{c}) \in \boldsymbol{HB}_{\pi(P)} \mid p(\boldsymbol{c}) \notin I\}, \text{ and}$$
$$\pi_2(I) = \{\pi_A \in \boldsymbol{HB}_{\pi(P)} \mid A \in \boldsymbol{DL}_P^- \,\&\, I \not\models_O A\}$$

With this in place, we establish a one-to-one mapping between the strong answer sets of a dl-program $\mathcal{K}$ and those of $\pi(\mathcal{K})$.

Theorem 1. *Let $\mathcal{K} = (O, P)$ be a dl-program. Then*

(i) if I is a strong answer set of $\mathcal{K}$, then $\pi(I)$ is a strong answer set of $\pi(\mathcal{K})$;
(ii) if I^* is a strong answer set of $\pi(\mathcal{K})$, then $I^* \cap HB_P$ is a strong answer set of $\mathcal{K}$.

Proof sketch: First one shows that $I \models_O DL[\lambda; Q](\boldsymbol{t})$ iff $\pi(I) \models_O DL[\pi(\lambda); Q](\boldsymbol{t})$. Then ones prove, by induction on k, that $\gamma^k_{\mathcal{K}^{s,I}} = HB_P \cap \gamma^k_{[\pi(\mathcal{K})]^{s,\pi(I)}}$ for every interpretation I and $k \geq 0$. Finally we can obtain that the least fixpoint of $\gamma_{[\pi(\mathcal{K})]^{s,\pi(I)}}$ is $\pi(I)$, thus (i) is proved. The proof of (ii) is similar. ■

Note that to remove the constraint operator from nonmonotonic dl-atoms, we must extend the underlying language. Indeed, the strong answer sets of an arbitrary dl-program $\mathcal{K}$ are not necessarily incomparable, while the strong answer sets of any $\cap$-free dl-program $\mathcal{K}'$ are minimal models of $\mathcal{K}'$ (cf. Theorem 4.13 of [11]) and thus incomparable. Hence, if $\mathcal{K}'$ results from a transformation that preserves strong answer sets, then it must use extra symbols.

Note that we need to determine the monotonicity of dl-atoms in the translation π, which leaves monotonic dl-atoms untouched. That is, the "double negation" interpretation applies only to positive nonmonotonic dl-atoms. If we deviate from this condition, the translation no longer works for strong answer sets. To illustrate this, we may ask whether monotonic dl-atoms can be handled like nonmonotonic dl-atoms, and if so, the translation turns out to be polynomial. Unfortunately we give a negative answer below.

Example 5. Consider $\mathcal{K}_1 = (\emptyset, P_1)$ where $P_1 = \{p(a) \leftarrow DL[S \uplus p, S' \cap q; S](a)\}$. The dl-atom $A = DL[S \uplus p, S' \cap q; S](a)$ is clearly monotonic and thus the unique strong answer set of $\mathcal{K}_1$ is $I = \emptyset$. If we extend π to eliminate the constraint operator from A similar as from nonmonotonic dl-atoms, we obtain the dl-program $(\emptyset, P_1')$ where

$$P_1' = \{p(a) \leftarrow not\, \pi_A, \quad \pi_A \leftarrow not\, DL[S \uplus p, S' \cup \pi_q; S](a), \quad \pi_q(a) \leftarrow not\, q(a)\ \}.$$

One can verify that this dl-program has two strong answer sets, $\{p(a), \pi_q(a)\}$ and $\{\pi_A, \pi_q(a)\}$, which are $\{p(a)\}$ and $\emptyset$ respectively when restricted to HB_{P_1}. Thus the modified translation may introduce strong answer sets that do not correspond to any strong answer set of the original dl-program.

Weak answer sets. Though we can show that Theorem 1 holds for $\mathcal{K}$ under weak answer set semantics as well, a similar translation exists for the purpose which needs not the knowledge about monotonicity of dl-atoms, thus it is polynomial. This is achieved by first rewriting dl-programs into ones in which all dl-atoms occur negatively.

Definition 3 ($\sigma(\mathcal{K})$)**.** *Let r be a dl-rule of the form (2). We define $\sigma(r)$ to be the dl-rule*

$$A \leftarrow \sigma(B_1), \ldots, \sigma(B_m), \ldots, not\, B_{m+1}, \ldots, not\, B_n$$

where $\sigma(B) = not\, \sigma_B$ if B is a dl-atom, and B otherwise, where σ_B is a fresh atom. For a dl-program $\mathcal{K} = (O, P)$, we define $\sigma(\mathcal{K}) = (O, \sigma(P))$ where $\sigma(P) = \{\sigma(r) \mid r \in P\} \cup \{\sigma_B \leftarrow not\, B \mid B \in DL_P\}$.

For instance, consider $\mathcal{K} = (\emptyset, P)$, where $P = \{p(a) \leftarrow DL[S \uplus p; S](a)\}$. We have $\sigma(\mathcal{K}) = (\emptyset, \sigma(P))$, where $\sigma(P) = \{p(a) \leftarrow not\, \sigma_A;\ \sigma_A \leftarrow not\, DL[S \uplus p; S](a)\}$, and A is the dl-atom $DL[S \uplus p; S](a)$. Then $\sigma(\mathcal{K})$ has two weak answer sets, viz. $\{\sigma_A\}$ and $\{p(a)\}$, which respectively correspond to the weak answer sets $\emptyset$ and $\{p(a)\}$ of $\mathcal{K}$ if restricted to HB_P. In general, we have:

Proposition 1. *Let $\mathcal{K} = (O, P)$ be a dl-program and $I \subseteq HB_P$. Then I is a weak answer set of $\mathcal{K}$ iff $I \cup \{\sigma_B \mid B \in DL_P \text{ and } I \not\models_O B\}$ is a weak answer set of $\sigma(\mathcal{K})$.*

Note that if in each rule of form (2) in a dl-program $\mathcal{K} = (O, P)$ every B_i $(1 \leq i \leq m)$ is an ordinary atom, then the weak answer sets of $\mathcal{K}$ coincide with the strong answer sets of $\mathcal{K}$. Now let $\pi'(\mathcal{K}) = \pi(\sigma(\mathcal{K}))$ be the composition of $\sigma(\mathcal{K})$ and $\pi(\mathcal{K})$. From Theorem 1 and Proposition 1, we thus obtain:

Corollary 1. *Let $\mathcal{K}$ be a dl-program. There is an one-one mapping between the weak answer sets of $\mathcal{K}$ and those of $\pi'(\mathcal{K}) = \pi(\sigma(\mathcal{K}))$.*

Clearly, $\sigma(\mathcal{K})$ can be computed in polynomial time w.r.t. the size of $\mathcal{K}$. Furthermore, in $\sigma(\mathcal{K})$ only negated dl-atoms occur, π needs no knowledge about monotonicity of dl-atoms for $\sigma(\mathcal{K})$, thus π' is a faithful and polynomial translation.

4 Complexity of Deciding Monotonicity of Dl-atoms

As regards monotonicity checking, we start with some generic result. Let us call the complexity of deciding, given a dl-program $\mathcal{K} = (O, P)$, a dl-atom A of form (1) in P, and an arbitrary interpretation I, whether $O(I; \lambda) \models A$ the *query complexity* of A.

Proposition 2. *Given a dl-program $\mathcal{K} = (O, P)$ and a dl-atom A occurring in P that has query complexity in class C, deciding whether A is monotonic is in* co-NP^C.

Indeed, to refute that A is monotonic, we can guess interpretations $I \subseteq I'$ and check that $(I; \lambda) \models A$ and $(I'; \lambda) \not\models A$ using the C oracle.

In the following, we study the complexity of monotonicity checking where O is from DL-Lite$_\mathcal{R}$ [5], $\mathcal{EL}^{++}$ [2], $\mathcal{SHIF}$ or $\mathcal{SHOIN}$ (the latter two were adopted in [11]).[9]

4.1 Description logic DL-Lite$_\mathcal{R}$

Starting from **I**, **C**, and **R**, *basic concepts* B and *(general) concepts* C, and *basic roles* R and *(general) roles* E are defined as as follows, where A is a concept name, P is a role name, and P^- is the *inverse* of P:

$$B \longrightarrow A \mid \exists R \qquad C \longrightarrow B \mid \neg B \qquad R \longrightarrow P \mid P^- \qquad E \longrightarrow R \mid \neg R$$

In a DL-Lite$_\mathcal{R}$ ontology $O = \langle \mathcal{T}, \mathcal{A} \rangle$, inclusion axioms are of the form $B \sqsubseteq C$ or $R \sqsubseteq E$, and membership assertions are of the form $C(a)$ and $E(a, b)$, where C is a concept, E is role and a, b are individual names.[10]

For every first-order sentence φ, we denote by $cls(\varphi)$ the clausal form of φ that is obtained by standard skolemization and transformation into set of clauses, and we let $cls(\Sigma) = \bigcup_{\varphi \in \Sigma} cls(\varphi)$. It is easy to see that if O is a DL-Lite$_\mathcal{R}$ ontology, every clause in $cls(\tau(O))$ contains at most two literals. Thus from the structure of resolution proofs, we obtain the following useful result (which is implicit in [5]).

[9] [11] considered the extensions $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$ of $\mathcal{SHIF}$ and $\mathcal{SHOIN}$, respectively, with datatypes; we omit the latter for simplicity (which is conceptually inessential).

[10] General concepts and roles in ABoxes is syntactic sugar; we can replace $C(a)$ and $E(a, b)$ by $A_C(a)$ and $P_E(a, b)$, resp., and add $A_C \sqsubseteq C$, $P_E \sqsubseteq E$ in the TBox, where A_C and P_E are fresh names. Modulo the new symbols, the new and the old ontology are equivalent.

Proposition 3. *Let $O = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL-Lite$_\mathcal{R}$ ontology. Then (i)–(iii) are equivalent:*

(i) O is unsatisfiable.
(ii) The empty clause has a resolution proof from $cls(\tau(O))$.
(iii) $\langle \mathcal{T}, \mathcal{A}' \rangle$ is unsatisfiable for some $\mathcal{A}' \subseteq \mathcal{A}$ with $|\mathcal{A}'| \leq 2$.

As well-known, deciding satisfiability of a DL-Lite$_\mathcal{R}$ knowledge base is tractable (cf. Theorem 26 of [5]); in fact, it is NLOG-complete [1]. Furthermore, subsumption checking (deciding $O \models F$ where F is an inclusion axiom) and instance checking (deciding $O \models F$ where F is a membership assertion) easily reduce to deciding knowledge base satisfiability for DL-Lite$_\mathcal{R}$ and are thus solvable in polynomial time.

This continues to hold if in F arbitrary conjunctions $X = X_1 \sqcap \cdots \sqcap X_n$ and disjunctions $Y = Y_1 \sqcup \cdots \sqcup Y_m$ of general concepts (resp. roles) X_i and Y_j are allowed in place of B and C (resp. R and E), Such compound queries, which are convenient for the user, can be reduced to basic reasoning using simple rewriting techniques, without the loss of tractability.

We thus define *dl-queries* in DL-Lite$_\mathcal{R}$ as formulas $Q(\boldsymbol{t})$ of one of the following forms, where $C^{(i)}$ (resp. $E^{(i)}$), $i = 1, 2$ are any conjunctions or disjunctions of general concepts (resp. roles), $B^{\sqcup}$ and $R^{\sqcup}$ are disjunctions of basic concepts resp. roles, and $C^{\sqcap}$ and $E^{\sqcap}$ are conjunctions of concepts resp. roles:

- $C^{(1)}(a)$, where $\boldsymbol{t} = a$, and a is a constant in $\mathcal{C}$;
- $E^{(1)}(a, b)$, where $\boldsymbol{t} = (a, b)$, a and b are constants in $\mathcal{C}$;
- $C^{(1)} \sqsubseteq C^{(2)}$, $\neg(B^{\sqcup} \sqsubseteq C^{\sqcap})$, $E^{(1)} \sqsubseteq E^{(2)}$, $\neg(R^{\sqcup} \sqsubseteq E^{\sqcap})$ where $\boldsymbol{t} = \epsilon$.

The entailment relation $O \models Q(\boldsymbol{t})$ is as in the discussion. Let the *size* of dl-query $Q(\boldsymbol{t})$, denoted by $|Q|$, be the number of concept resp. role names occurring in Q. As a consequence of Proposition 3, we show:

Proposition 4. *Let $O = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL-Lite$_\mathcal{R}$ knowledge base and $Q(\boldsymbol{t})$ a dl-query.*

(i) Deciding whether $O \models Q(\boldsymbol{t})$ is feasible in polynomial time of the size O (more precisely, NLOG*-complete).*
(ii) $O \models Q(\boldsymbol{t})$ iff $\langle \mathcal{T}, \mathcal{A}' \rangle \models Q(\boldsymbol{t})$ for some $\mathcal{A}' \subseteq \mathcal{A}$ and $|\mathcal{A}'| \leq \max(2, |Q|^2)$.

Regarding monotonicity checking, we then prove the following result.

Theorem 2. *Given a dl-program $\mathcal{K} = (O, P)$, where O is a DL-Lite$_\mathcal{R}$ ontology, and a dl-atom A occurring in P, deciding whether A is monotonic is* co-NP*-complete.*

Proof sketch: The co-NP membership follows from Prop. 2 and the fact the query complexity of the dl-queries $Q(\boldsymbol{t})$ above is polynomial (Prop. 4). For co-NP-hardness, we give a reduction from 3SAT instances $\varphi = \varphi_1 \wedge \cdots \wedge \varphi_m$ with variables $x_1, \ldots, x_k$, and $\varphi_i = l_{i,1} \vee l_{i,2} \vee l_{i,3}$. We construct the ontology O and the dl-atom A as follows:

- $O = \{l^*_{i,j} \sqsubseteq B_i \mid 1 \leq i \leq m, 1 \leq j \leq 3\}$ with $x^*_s = A_s$, and $\neg x^*_s = A'_s$, $1 \leq s \leq k$,
- $A = DL[\lambda; B \sqcap B_1 \sqcap \cdots \sqcap B_m](d)$ with $\lambda = (A_1 \uplus p_1, \ldots, A_k \uplus p_k, \neg A'_1 \cap p_1, \ldots, \neg A'_k \cap p_k, \neg B \cap p)$,

where (i) $A_1, A'_1 \ldots, A_k, A'_k, B, B_1, \ldots, B_m$ are pairwise distinct atomic concepts, (ii) $p, p_1, \ldots, p_k$ are pairwise distinct unary predicates, and (iii) d is a constant. Finally we can show that φ is satisfiable iff A is nonmonotonic relative to O. ■

Note that in the proof of co-NP-hardness, the query size $|Q|$ grows with the size of the 3SAT instance. We next show that monotonicity checking is tractable if $|Q|$ is small (bounded by a constant); this is perhaps the most frequent case in practice.

The proof exploits two key properties; the first is that satisfaction of a dl-atom can be shown with small interpretations relative to the query size.

Proposition 5. *Let $\mathcal{K} = (O, P)$ be a dl-program, $I \subseteq HB_P$ and A a dl-atom of form (1) from P. If $I \models_O A$, then $I' \models_O A$ for some $I' \subseteq I$ such that $|I'| \leq \max(2, |Q|^2)$.*

In case $|Q| = 1$, $|I'| = 2$ can be necessary. For example, consider the dl-atom $A = DL[S_1 \uplus p, S_2 \uplus q; S_3](a)$ and $O = \{S_1 \sqsubseteq \neg S_2\}$, where S_1, S_2, S_3 are concept names. Then $\{p(a), q(a)\} \models_O A$ but no proper subset of $\{p(a), q(a)\}$ satisfies A under O.

The second property is that the monotonicity of a dl-atom can be refuted by changing a single atom in an interpretation I.

Proposition 6. *Let $\mathcal{K} = (O, P)$, where O be a (DL-Lite$_\mathcal{R}$) ontology. A dl-atom A from P is nonmonotonic iff there exists $I \subseteq HB_P$ and $p(\boldsymbol{c}) \in HB_P$ such that $I \models_O A$ and $I \cup \{p(\boldsymbol{c})\} \not\models_O A$.*

Using Propositions 5 and 6, we establish the announced tractability result.

Theorem 3. *Given a dl-program $\mathcal{K} = (O, P)$, where O is a DL-Lite$_\mathcal{R}$ ontology, and a dl-atom $A = DL[\lambda; Q](\boldsymbol{t})$ from P of the form (1), deciding whether A is monotonic is feasible in polynomial time (more precisely,* NLOG*-complete) if the size of $Q(\boldsymbol{t})$ is bounded by a constant.*

Proof sketch: Let $n = |Q|$ and $Q(\boldsymbol{t}) = C(a)$ where C is a concept name. Clearly $O \not\models C(a)$, otherwise A is monotonic. By Proposition 6, A is nonmonotonic iff there exist some $I \subseteq HB_P$, $p(\boldsymbol{c}) \notin I$ such that (1) $I \models_O A$ and (2) $I \cup \{p(\boldsymbol{c})\} \not\models_O A$ iff $P^+ \cap P^- = \emptyset$ and some I' of size $|I'| \leq \max(n^2, 2)$ exists such that $I' \cap P^+ = \emptyset$ and $I' \models_O A$, where $P^s = \{\alpha \in HB_P \mid O \cup A(\{\alpha\}) \models C(a)\}$, for $s \in \{+, -\}$ by Proposition 3. As easily seen, $p(\boldsymbol{c}) \in P^-$. Now the sets P^+ and P^- can be computed in polynomial time. If n is bounded by a constant, the search space for I' is polynomial; the test $I' \models_O A$ is feasible in polynomial time. Thus deciding whether A is nonmonotonic is feasible in polynomial time. Compound dl-queries can be reduced to the above case by formula rewriting techniques. ■

4.2 Description logic $\mathcal{EL}^{++}$

The $\mathcal{EL}^{++}$ *concepts* are defined by

$$C, D ::= \top \mid \bot \mid A \mid \{a\} \mid C \sqcap D \mid \exists P.C,$$

where $A \in \mathbf{C}$, $P \in \mathbf{R}$ and $a \in \mathbf{I}$. An $\mathcal{EL}^{++}$ *constraint box (CBox)*, or *knowledge base*, is a finite set of *general concept inclusions (GCIs)* of the form $C \sqsubseteq D$ and *role inclusions (RIs)* of the form $P_1 \circ \cdots \circ P_n \sqsubseteq P$. One can see that $\mathcal{EL}^{++}$ is expressive enough

to capture concept and role assertions, e.g. assertions $C(a)$ and $P(a,b)$ can be encoded with general concept inclusions $\{a\} \sqsubseteq C$ and $\{a\} \sqsubseteq \exists P.\{b\}$ respectively. Similarly, the assertions $\neg C(a)$ and $\neg P(a,b)$ can be encoded as $\{a\} \sqcap C \sqsubseteq \bot$ and $\{a\} \sqcap \exists P.\{b\} \sqsubseteq \bot$ respectively. It is known that subsumption in $\mathcal{EL}^{++}$ w.r.t. CBoxes is tractable (cf. Theorem 4 of [2]), and the same holds for satisfiability in $\mathcal{EL}^{++}$ w.r.t. CBoxes.

By an *extended concept* we mean a concept C or its negation $\neg C$, and an *extended role* we mean a role P or its negation $\neg P$. An *dl-query* $Q(\boldsymbol{t})$ in $\mathcal{EL}^{++}$ has one of the following forms, where $C^{(i)}$ (resp. $E^{(i)}$), $i = 1, 2$ are any conjunctions or disjunctions of extended concepts (resp. roles), $C^{\sqcup}$ and $E^{\sqcup}$ are disjunctions of concepts resp. roles, and $C^{\sqcap}$ and $E^{\sqcap}$ are conjunctions of extended concepts resp. of roles:

- $C^{(1)}(a)$, where $\boldsymbol{t} = a$, and a is a constant in $\mathcal{C}$;
- $E^{(1)}(a,b)$, where $\boldsymbol{t} = (a,b)$, a and b are constants in $\mathcal{C}$;
- $C^{(1)} \sqsubseteq C^{(2)}$, $\neg(C^{\sqcup} \sqsubseteq C^{\sqcap})$, $E^{(1)} \sqsubseteq E^{(2)}$, $\neg(E^{\sqcup} \sqsubseteq E^{\sqcap})$, where $\boldsymbol{t} = \epsilon$.

Then we have the following property.

Proposition 7. *Given an $\mathcal{EL}^{++}$ knowledge base O and a dl-query $Q(\boldsymbol{t})$, deciding whether $O \models Q(\boldsymbol{t})$ is feasible in polynomial time.*

Dl-atoms have the same form (1) except that concepts and roles occurring in λ are now in $\mathcal{EL}^{++}$. It is then not hard to establish that they have polynomial query complexity. Thus from Proposition 2 and a slight modification of the reduction in the proof of Theorem 2, we obtain the following result.

Theorem 4. *Given a dl-program $\mathcal{K} = (O, P)$, where O is an $\mathcal{EL}^{++}$ ontology, and a dl-atom $A = DL[\lambda; Q](\boldsymbol{t})$ in DL_P, deciding whether A is monotonic is* co-NP*-complete. The problem is* co-NP*-hard even if $Q(\boldsymbol{t}) = C(a)$ where C is a concept name.*

4.3 Description logics $\mathcal{SHIF}$ and $\mathcal{SHOIN}$

The description logics $\mathcal{SHIF}$ and $\mathcal{SHOIN}$ subsume DL-Lite$_{\mathcal{R}}$ and $\mathcal{EL}^{++}$. They allow for transitive roles, i.e., role inclusion axioms of the form $r \circ r \sqsubseteq r$, and role inclusion $r \sqsubseteq s$, under syntactic restrictions to ensure decidability of the basic reasoning tasks. Furthermore, they cater equality $a \approx b$ and inequality $a \not\approx b$ of individuals a and b in ABox assertions. $\mathcal{SHOIN}$ concepts can be built inductively using the Boolean connectives, nominal concepts $\{o_1, \ldots, o_n\}$, qualified restrictions $\exists R.Q, \forall R.Q$ and number restrictions $\geq nR$, and $\leq nR$, where n is an integer; in $\mathcal{SHIF}$, nominal concepts are excluded and $\geq nR$ is restricted to $\geq 1R$ and $\leq nR$ to $\leq 0R$; see [15] for details.

Dl-atoms have the same form as (1) except that

- each S_i is either a concept, a $\mathcal{SHOIN}$ resp. $\mathcal{SHIF}$ role or its negation,[11] or one of $\approx$, $\not\approx$;
- the dl-query $Q(\boldsymbol{t})$ has the form in Section 2.2 (using $\mathcal{SHOIN}$ / $\mathcal{SHIF}$ concepts and roles) with role conjunctions resp. disjunctions of size 1, or the form $t_1 \approx t_2$ or $t_1 \not\approx t_2$ where $\boldsymbol{t} = (t_1, t_2)$.

[11] It is convenient to allow for role negation, as we can replace "$S \uplus p$" equivalently with "$\neg S \uplus p$". As discussed in [11], negative role assertions can be simulated in $\mathcal{SHIF}$ and $\mathcal{SHOIN}$.

Recall that testing knowledge base satisfiability is EXP-complete for $\mathcal{SHIF}$ [24,15] and NEXP-complete for $\mathcal{SHOIN}$ (for both unary and binary number encoding) [15,19]. For monotonicity checking of dl-atoms, we establish the following result.

Theorem 5. *Given a dl-program $\mathcal{K} = (O, P)$ and a dl-atom A occurring in P, deciding whether A is monotonic is (i)* EXP*-complete, if O is a $\mathcal{SHIF}$ knowledge base and (ii)* P^{NEXP}*-complete, if O is a $\mathcal{SHOIN}$ knowledge base.*

Proof sketch: The membership parts follow easily from Proposition 2 and the facts that (i) $\mathcal{SHIF}$ has query complexity in EXP and co-$\text{NP}^{\text{EXP}} = \text{EXP}$, and (ii) $\mathcal{SHOIN}$ has query complexity in co-NEXP and co-$\text{NP}^{\text{co-NEXP}} = \text{P}^{\text{NEXP}}$ (cf. [11]). For the EXP-hardness of (i), let O be a $\mathcal{SHIF}$ knowledge base, and define $\mathcal{K} = (O', P)$ where $O' = O \cup \{C(o) \mid o \in \mathcal{C}\}$, $P = \{p^* \leftarrow \mathit{DL}[C \uplus p; \top \sqsubseteq \bot]\}$, where C is a fresh concept name. Then the dl-atom in P is monotonic relative to $\mathcal{K}$ iff O is unsatisfiable, which is EXP-complete in general (as EXP is closed under complementation). The P^{NEXP}-hardness of (ii) is proved by a polynomial transformation of the following problem NEXP-JC, which is P^{NEXP}-complete, to dl-atom monotonicity checking. ■

Thus testing monotonicity-of a dl-atom is not or only mildly harder than deciding knowledge base satisfiability; it is in fact as hard as deciding whether a dl-program has a weak or strong answer set over the respective description logic.

5 Discussion and Conclusion

5.1 Related work

To the best of our knowledge, eliminating the constraint operator from dl-programs has been considered by Eiter *et al.* [13], who proposed a well-founded semantics for dl-programs mentioning no the constraint operator and argued that the well-founded semantics for general dl-programs is obtained by translating them into ones without constraint operators. Our transformation π for the strong answer set semantics is inspired by the respective transformation in [13] (which we refer to as π' in the following), but there are subtle differences which result in significantly different behavior.

For every dl-program $\mathcal{K} = (O, P)$, we have $\pi'(\mathcal{K}) = (O, \pi'(P))$ where $\pi'(P) = \bigcup_{r \in P} \pi'(r)$ and $\pi'(r)$ comprises the following rules:

(1) if $S \uplus p$ occurs in r, then $\pi'(r)$ includes all ground instances of the rule

$$\overline{p}(\boldsymbol{X}) \leftarrow \mathit{not}\ \mathit{DL}[S' \uplus p; S'](\boldsymbol{X}),$$

where S' is a fresh concept (resp., role) name if p is a unary (resp., binary) predicate, and $\boldsymbol{X}$ is a tuple of distinct variables matching the arity of p,

(2) $\pi'(r)$ includes the rule obtained from r by replacing each "$S \uplus p$" with "$\neg S \uplus \overline{p}$".[12] For every atom or dl-atom A, let $\pi'(A)$ result from A by replacing every occurrence $S \uplus p$ with $\neg S \uplus \overline{p}$.

The translation π' may lose strong answer sets as demonstrated in the next example.

[12] It is "$S \uplus \overline{p}$" according to [13] which is equivalent to "$\neg S \uplus \overline{p}$".

Example 6. Consider the dl-program $\mathcal{K} = (O, P)$ where $O = \emptyset$ and P consists of

$$q(a) \leftarrow \mathit{DL}[S_1 \uplus p, \neg S_2 \cap q; S_1 \sqcup S_2](a), \qquad p(a) \leftarrow q(a).$$

It is not difficult to check that $\mathcal{K}$ has a unique strong answer set $\{p(a), q(a)\}$. One can verify that $\pi'(\mathcal{K})$ has no strong answer sets.

The discussion above leads to a related question, viz. whether the translation $\pi'(\mathcal{K})$ introduces extra strong answer sets. However, this is not the case.

Proposition 8. *Let $\mathcal{K} = (O, P)$ be a dl-program and let $I \subseteq HB_{\pi'(P)}$ be a strong answer set of $\pi'(\mathcal{K})$. Then $I \cap HB_P$ is a strong answer set of $\mathcal{K}$.*

As shown above, dl-programs can be faithfully transformed into canonical dl-programs, i.e., dl-programs without nonmonotonic dl-atoms, under both the strong and the weak answer sets semantics. Besides the use for computational purposes in solvers for dl-programs, the transformation for strong answer sets enables us to show that arbitrary dl-programs under this semantics can be embedded into hybrid Minimal Knowledge and Negation as Failure (MKNF) [18] and First-Order Autoepistemic Logic (FO-AEL) [7], generalizing respective embeddings for canonical dl-programs in these papers, and into default logic [20], which will be reported elsewhere.

5.2 Open issues

Our complexity results on monotonicity testing of dl-atoms suggest that a polynomial and faithful translation of arbitrary dl-programs into canonical ones under strong answer sets semantics, by eliminating the nonmonotonic dl-atoms is unlikely to exist for DL-Lite$_\mathcal{R}$ and $\mathcal{EL}^{++}$, as the existence of a strong answer set has NP complexity while monotonicity testing is co-NP-complete. However, the two problems have the same complexity for $\mathcal{SHIF}$ and $\mathcal{SHOIN}$, and the respective complexity classes are closed under complement. This suggests that it may be possible to transform general dl-programs into equivalent canonical ones under the strong answer sets semantics by eliminating constraint operators from dl-atoms in polynomial time. An idea is that in the transformation $\pi(\mathcal{K})$, the monotonicity test for $\pi(B)$ in the rules (4) can be encoded to a designated dl-program that is incorporated into the given dl-program and calculates the test result. However, this requires further investigation.

Besides the weak and strong answer set semantics, further notions of answer set semantics for dl-programs have been proposed that refine the weak and strong answer set semantics, including FLP semantics [12,14] and the recent well-supported semantics [23]. It remains an interesting issue to investigate whether translations similar to $\pi(\mathcal{K})$ for dl-programs under these answer set semantics to exist.

Acknowledgment. This work was supported by the Austrian Science Fund (FWF) project P20840. Yisong Wang was partially supported by NSFC grant 60963009, Open Funds of the State Key Laboratory of Computer Science of Chinese Academy of Science grant SYSKF1106 and Stadholder Fund of Guizhou Province grant (2012)62.

References

1. Alessandro Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The DL-Lite family and relations. *Journal of Artifical Intelligence Research*, 36:1–69, 2009.

2. Franz Baader, S. Brandt, and C. Lutz. Pushing the EL envelope. In *IJCAI-2005*, pp. 364–369.
3. Franz Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider. *The Description Logic Handbook*. Cambridge Univ. Press, 2nd ed., 2007.
4. Gerhard Brewka, T. Eiter, and M. Truszczynski. Answer set programming at a glance. *Communications of the ACM*, 54(12):92–103, 2011.
5. Diego Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *Journal of Automated Reasoning*, 39(3):385–429, 2007.
6. Jos de Bruijn, Philippe Bonnard, Hugues Citeau, Sylvain Dehors, Stijn Heymans, Jörg Pührer, and Thomas Eiter. Combinations of rules and ontologies: State-of-the-art survey of issues. Technical Report Ontorule D3.1, Ontorule Project Consortium, 2009. `http://ontorule-project.eu/`.
7. Jos de Bruijn, T. Eiter, and H. Tompits. Embedding approaches to combining rules and ontologies into autoepistemic logic. In *KR-2008*, pp. 485–495, 2008. AAAI Press.
8. Jos de Bruijn, D. Pearce, A. Polleres, and A. Valverde. Quantified equilibrium logic and hybrid rules. In *RR-2007*, *LNCS* 4524, pp. 58–72, 2007. Springer.
9. Thomas Eiter, M. Fink, T. Krennwallner, C. Redl, and P. Schüller. Exploiting unfounded sets for hex-program evaluation. In *JELIA*, *LNCS* 7519, pp. 160–175, 2012. Springer.
10. Thomas Eiter, M. Fink, and D. Stepanova. Semantic independence in dl-programs. In *RR-2012*, *LNCS* 7497, pp. 58–74. Springer, 2012.
11. Thomas Eiter, G. Ianni, T. Lukasiewicz, R. Schindlauer, and H. Tompits. Combining answer set programming with description logics for the semantic web. *Artificial Intelligence*, 172(12-13):1495–1539, 2008.
12. Thomas Eiter, G. Ianni, R. Schindlauer, and H. Tompits. A uniform integration of higher-order reasoning and external evaluations in answer-set programming. In *IJCAI-2005*, pp. 90–96, 2005. Professional Book Center.
13. Thomas Eiter, T. Lukasiewicz, G. Ianni, and R. Schindlauer. Well-founded semantics for description logic programs in the semantic web. *ACM TOCL*, 12(2):11:1–11:41, 2011.
14. Michael Fink and D. Pearce. A logical semantics for description logic programs. In *JELIA-2010*, *LNCS* 6341, pp. 156–168, 2010. Springer.
15. Ian Horrocks and P. F. Patel-Schneider. Reducing OWL entailment to description logic satisfiability. In *ISWC-2003*, *LNCS* 2870, pp. 17–29, 2003. Springer.
16. Joohyung Lee and R. Palla. Integrating rules and ontologies in the first-order stable model semantics (preliminary report). In *LPNMR-2011*, *LNCS* 6645, pp. 248–253, 2011. Springer.
17. Vladimir Lifschitz, L. R. Tang, and H. Turner. Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, 25(3-4):369–389, 1999.
18. Boris Motik and R. Rosati. Reconciling description logics and rules. *Journal of the ACM*, 57(5):1–62, 2010.
19. Ian Pratt-Hartmann. Complexity of the two-variable fragment with counting quantifiers. *Journal of Logic, Language and Information*, 14(3):369–395, 2005.
20. Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13(1-2):81–132, 1980.
21. Riccardo Rosati. On the decidability and complexity of integrating ontologies and rules. *Journal of Web Semantics*, 3(1):61–73, 2005.
22. Riccardo Rosati. DL+log: Tight integration of description logics and disjunctive datalog. In *KR-2006*, pp. 68–78, 2006. AAAI Press.
23. Yi-Dong Shen. Well-supported semantics for description logic programs. In *IJCAI-2011*, pp. 1081–1086, 2011. IJCAI/AAAI.
24. Stephan Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. PhD thesis, RWTH Aachen, Germany, 2001.
25. Yisong Wang, J.-H. You, L. Y. Yuan, Y.-D. Shen, and M. Zhang. The loop formula based semantics of description logic programs. *TCS*, 415:60–85, 2012.