# Well-Supported Semantics for Description Logic Programs

**Yi-Dong Shen**
State Key Laboratory of Computer Science,
Institute of Software, Chinese Academy of Sciences
Beijing 100190, China
ydshen@ios.ac.cn

## Abstract

[Fages, 1994] introduces the notion of well-supportedness as a key requirement for the semantics of normal logic programs and characterizes the standard answer set semantics in terms of the well-supportedness condition. With the property of well-supportedness, answer sets are guaranteed to be free of circular justifications. In this paper, we extend Fages' work to description logic programs (or DL-programs). We introduce two forms of well-supportedness for DL-programs. The first one defines *weakly well-supported* models that are free of circular justifications caused by positive literals in rule bodies. The second one defines *strongly well-supported* models that are free of circular justifications caused by either positive or negative literals. We then define two new answer set semantics for DL-programs and characterize them in terms of the weakly and strongly well-supported models, respectively. The first semantics is based on an extended Gelfond-Lifschitz transformation and defines *weakly well-supported answer sets* that are free of circular justifications for the class of DL-programs without negative dl-atoms. The second semantics defines *strongly well-supported answer sets* which are free of circular justifications for all DL-programs. We show that the existing answer set semantics for DL-programs, such as the weak answer set semantics, the strong answer set semantics, and the FLP-based answer set semantics, satisfy neither the weak nor the strong well-supportedness condition, even for DL-programs without negative dl-atoms. This explains why their answer sets incur circular justifications.

## 1 Introduction

Description logic programs (or DL-programs) are introduced in [Eiter *et al.*, 2008b] as a framework for integrating logic programming under the answer set semantics (ASP) [Gelfond and Lifschitz, 1991] with description logics (DLs) [Baader *et al.*, 2003] for the Semantic Web. ASP is currently a dominating logic programming paradigm, while DLs like $\mathcal{SHIF}(D)$ and $\mathcal{SHOIN}(D)$ are the semantical basis of the Web Ontology Language OWL [Horrocks *et al.*, 2003]. A DL-program $KB = (L, R)$ consists of a DL knowledge base $L$ expressing ontologies and an ASP rule base $R$ expressing constraints. $L$ and $R$ are separated in syntax in that they share no predicate symbols in their vocabularies. The exchange of knowledge between the two components is through an interface mechanism in rule bodies, called *dl-atoms*, which are interpreted as queries to $L$. It is due to the syntactical separation of $L$ and $R$ that DL-programs are classified as a *loosely coupled* integration, which is entirely different from the *tightly coupled* or *full* integrations such as [de Bruijn *et al.*, 2008; Lukasiewicz, 2010; Motik and Rosati, 2010; Rosati, 2006] (see [Eiter *et al.*, 2008a] for a survey).

In [Eiter *et al.*, 2008b], two different semantics are proposed for DL-programs. One is called the *weak answer set semantics*, and the other the *strong answer set semantics*. Let $KB = (L, R)$ be a DL-program and $I$ an interpretation. Informally, $I$ is a weak answer set of $KB$ if $I$ is the least model of the transformed logic program $wR_L^I$ which is obtained from $ground(R)$ (the grounding of $R$) by first removing all rules whose bodies are not satisfied by $I$, then removing all negative literals and dl-atoms from the remaining rules. [Eiter *et al.*, 2008b] note that an obvious disadvantage of the weak answer set semantics is that it may produce counterintuitive answer sets with circular justifications by self-supporting loops. The circular justifications are caused by some dl-atoms in rule bodies, but these dl-atoms are completely ignored (deleted) in $wR_L^I$. In order to overcome this problem, [Eiter *et al.*, 2008b] further introduce the strong answer set semantics, which is different from the weak one in that only *nonmonotonic* dl-atoms are deleted in the transformation. A dl-atom $A$ is *monotonic* relative to $KB$ if for any interpretations $I$ and $J$ with $I \subseteq J$, if $I$ satisfies $A$ then $J$ satisfies $A$. $I$ is a strong answer set of $KB$ if $I$ is the least model of the transformed logic program $sR_L^I$ obtained from $ground(R)$ by first removing all rules whose bodies are not satisfied by $I$, then removing all negative literals and nonmonotonic dl-atoms from the remaining rules.

It turns out that the strong answer set semantics cannot overcome the problem of circular justifications. Some circular justifications are caused by nonmonotonic dl-atoms, but these dl-atoms are all deleted and thus ignored in $sR_L^I$.

For normal logic programs, the problem of circular jus-

tifications is elegantly handled by means of Fages' *well-supportedness* [Fages, 1994]. Informally, an interpretation $I$ is well-supported if there is a level mapping on $I$ such that for every $a \in I$, there is a rule in $ground(R)$ of the form $a \leftarrow b_1, \cdots, b_m, not\ c_1, \cdots, not\ c_n$ such that $I$ satisfies the body of the rule and for each $b_i$, the level of $b_i$ is lower than the level of $a$. By enforcing the well-supportedness condition on interpretations, answer sets of normal logic programs are guaranteed to be free of circular justifications.

Fages' definition of well-supportedness does not apply to DL-programs, since it handles circular justifications caused only by positive literals (atoms) in rule bodies. The question naturally arises: can we extend Fages' well-supportedness from normal logic programs to DL-programs and define a well-supported answer set semantics for DL-programs such that its answer sets are free of circular justifications? Definitely, the extension is nontrivial and challenging because circular justifications in a DL-program are caused either by positive literals (atoms or dl-atoms) or negative literals (negative atoms or negative dl-atoms) in rule bodies.

In this paper, we extend Fages' well-supportedness to DL-programs. In particular, we introduce two forms of well-supportedness for DL-programs. The first one defines what we call *weakly well-supported* models, which are free of circular justifications caused by positive literals in rule bodies. The second one defines *strongly well-supported* models, which are free of circular justifications caused by either positive or negative literals. The two forms of well-supportedness are proper extensions of Fages' well-supportedness in that when applied to a DL-program $KB = (L, R)$ that represents a normal logic program (i.e., $L$ is empty and $R$ contains no dl-atoms), they coincide with Fages' well-supportedness. We show that the weak answer set semantics and the strong answer set semantics of [Eiter *et al.*, 2008b] satisfy neither the weak nor the strong well-supportedness condition, even for DL-programs without negative dl-atoms. This explains why their answer sets incur circular justifications.

We then define two new answer set semantics for DL-programs and characterize them in terms of the weakly and strongly well-supported models, respectively. The first semantics is based on an extended Gelfond-Lifschitz transformation and defines what we call *weakly well-supported answer sets*. Such answer sets are free of circular justifications for the class of DL-programs without negative dl-atoms. The second semantics does not rely on program transformations and defines *strongly well-supported answer sets* which are free of circular justifications for all DL-programs. We show that strongly well-supported answer sets are weakly well-supported answer sets that are also strong answer sets.

## 2 DL-Programs

We follow the notation of [Eiter *et al.*, 2008b] and assume familiarity with the basics of description logics (DLs) [Baader *et al.*, 2003], especially with the DLs $\mathcal{SHIF}(D)$ and $\mathcal{SHOIN}(D)$ [Horrocks *et al.*, 2003]. A *DL knowledge base* $L$ consists of a finite set of axioms built over a vocabulary $\Sigma_L = (\mathbf{A} \cup \mathbf{R}, \mathbf{I})$, where $\mathbf{A}$, $\mathbf{R}$ and $\mathbf{I}$ are pairwise disjoint (denumerable) sets of atomic concepts, atomic roles and individuals, respectively. DLs are fragments of first-order logic, so the DL knowledge base $L$ has the first-order semantics.

Let $\mathbf{P}$ be a finite set of predicate symbols and $\mathbf{C}$ a nonempty finite set of constants such that $\mathbf{P} \cap (\mathbf{A} \cup \mathbf{R}) = \emptyset$ and $\mathbf{C} \subseteq \mathbf{I}$. A term is either a constant from $\mathbf{C}$ or a variable. An atom is of the form $p(t_1, ..., t_m)$, where $p$ is a predicate symbol from $\mathbf{P}$, and $t_i$ is a term. An equality (resp. inequality) is of the form $t_1 = t_2$ (resp. $t_1 \neq t_2$), where $t_1$ and $t_2$ are terms. A *dl-query* is of the form $Q(\mathbf{t})$, where $\mathbf{t}$ is a list of terms, and $Q$ is an equality/inequality symbol, a concept, a role, a concept inclusion axiom, or their negation, built from $\mathbf{A} \cup \mathbf{R}$.

A *dl-atom* is of the form $DL[S_1 op_1 p_1, \cdots, S_m op_m p_m; Q](\mathbf{t})$, where $S_i$ is a concept or role built from $\mathbf{A} \cup \mathbf{R}$, or an equality/inequality symbol; $op_i \in \{ \uplus, \uplus, \cap \}$ is an operator; $p_i \in \mathbf{P}$ is a unary predicate symbol if $S_i$ is a concept, and a binary predicate symbol otherwise; and $Q(\mathbf{t})$ is a dl-query. The semantics of each operator $op_i$ shall be defined in Definition 1 below.

A *dl-rule* (or rule) $r$ is of the form

$$H \leftarrow A_1, \cdots, A_m, not\ B_1, \cdots, not\ B_n \qquad (1)$$

where $H$ is an atom, and the $A_i$s and $B_i$s are atoms or equalities/inequalities or dl-atoms. Each $A_i$ is called a *positive literal*, and each $not\ B_i$ called a *negative literal*. We use $head(r)$ and $body(r)$ to denote the head $H$ and the body $A_1, \cdots, A_m, not\ B_1, \cdots, not\ B_n$, respectively. We also use $pos(r)$ to denote the positive literals $A_1, \cdots, A_m$, and $neg(r)$ to denote the negative literals $not\ B_1, \cdots, not\ B_n$. Therefore, a rule $r$ can simply be written as $head(r) \leftarrow body(r)$ or $head(r) \leftarrow pos(r), neg(r)$.

A *rule base* $R$ is a finite set of rules. $R$ is a *normal logic program* if it is free of dl-atoms, equalities and inequalities.

A *DL-program* is a combined knowledge base $KB = (L, R)$, where $L$ is a DL knowledge base and $R$ is a rule base.

A *ground instance* of a rule $r$ is obtained by first replacing every variable in $r$ with a constant from $\mathbf{C}$, then removing all valid equalities and inequalities (under the unique name assumption). A ground instance of $r$ is *consistent* if it contains no equalities or inequalities. Let $ground(R)$ denote the set of all consistent ground instances of rules in $R$.

The *Herbrand base* of $R$, denoted $HB_R$, is the set of all ground atoms $p(t_1, ..., t_m)$, where $p \in \mathbf{P}$ occurs in $R$ and each $t_i$ is in $\mathbf{C}$. Any subset of $HB_R$ is an *interpretation*. For an interpretation $I$, let $I^- = HB_R \setminus I$ and $\neg I^- = \{\neg a \mid a \in I^-\}$.

For a normal logic program $R$, an interpretation $I$ *satisfies* an atom $a \in HB_R$ if $a \in I$, and $I$ satisfies $not\ a$ if $a \notin I$. The satisfaction is extended to DL-programs as follows.

**Definition 1** Let $KB = (L, R)$ be a DL-program and $I$ an interpretation. Define the *satisfaction relation under* $L$, denoted $\models_L$, as follows:

1. For a ground atom $a \in HB_R$, $I \models_L a$ if $a \in I$.
2. For a ground dl-atom $A = DL[S_1 op_1 p_1, \cdots, S_m op_m p_m; Q](\mathbf{c})$ occurring in $ground(R)$, $I \models_L A$ if $L \cup \bigcup_{i=1}^{m} A_i \models Q(\mathbf{c})$, where $\models$ is the *entailment relation*

in first-order logic and

$$A_i = \begin{cases} \{S_i(\mathbf{e}) \mid p_i(\mathbf{e}) \in I\}, & \text{if } op_i = \uplus; \\ \{\neg S_i(\mathbf{e}) \mid p_i(\mathbf{e}) \in I\}, & \text{if } op_i = \uplus; \\ \{\neg S_i(\mathbf{e}) \mid p_i(\mathbf{e}) \notin I\}, & \text{if } op_i = \cap. \end{cases}$$

For a ground atom or dl-atom $A$, $I \models_L not\ A$ if $I \not\models_L A$. For a rule $r$ in $ground(R)$, $I \models_L body(r)$ if for each (positive or negative) literal $l$ in $body(r)$, $I \models_L l$; $I \models_L r$ if $I \not\models_L body(r)$ or $I \models_L head(r)$. An interpretation $I$ is a *model* of $KB$ if $I \models_L r$ for all $r \in ground(R)$. A *minimal model* is a model that is minimal in terms of set inclusion.

A ground dl-atom $A$ is *monotonic* relative to $KB$ if for any $I \subseteq J \subseteq HB_R$, $I \models_L A$ implies $J \models_L A$. Otherwise, $A$ is *nonmonotonic*. $KB$ is *positive* if $R$ is free of the negation symbol $not$, and that every dl-atom occurring in $ground(R)$ is monotonic relative to $KB$. A positive DL-program $KB$ has the least model.

For an interpretation $I$, let $sR_L^I$ be the *reduct* obtained from $ground(R)$ by deleting (i) every rule $r$ with $I \not\models_L body(r)$, and (ii) from the remaining rules all negative literals and all nonmonotonic dl-atoms. Let $wR_L^I$ be obtained in the same way as $sR_L^I$ except that in (ii) all negative literals and all dl-atoms are deleted.

[Eiter *et al.*, 2008b] define $I$ to be a *strong* (resp. *weak*) answer set of $KB$ if $I$ is the least model of $sR_L^I$ (resp. $wR_L^I$).

## 3 Well-Supported Semantics

For some DL-programs, applying the weak or strong answer set semantics may produce answer sets with circular justifications. For the weak answer set semantics, this problem is illustrated in [Eiter *et al.*, 2008b] with an example DL-program $KB = (L, R)$, where $L = \emptyset$ and $R = \{p(a) \leftarrow DL[c \uplus p; c](a)\}$. This DL-program has two weak answer sets: $I_1 = \emptyset$ and $I_2 = \{p(a)\}$. Let us use the symbol $\Leftarrow$ to express "truth is supported by". Then $p(a)$ in $I_2$ is circularly justified by a self-supporting loop: $p(a) \Leftarrow DL[c \uplus p; c](a) \Leftarrow p(a)$.

The next DL-program illustrates that for the strong answer set semantics, the circular justification problem persists.

**Example 1** Let $KB = (L, R)$, where $L = \emptyset$ and $R = \{p(a) \leftarrow DL[c \uplus p, b \cap q; c \sqcap \neg b](a)\}$. The dl-atom in $R$ queries $L$ if $a$ is in the concept $c$ but not in the concept $b$, under the assumption that for any $x$, if $p(x)$ is true then $x$ is in $c$ and if $q(x)$ is false then $x$ is not in $b$. This dl-atom is nonmonotonic, so both $I_1 = \emptyset$ and $I_2 = \{p(a)\}$ are strong answer sets of $KB$. Observe that $p(a)$ in $I_2$ is circularly justified by a self-supporting loop: $p(a) \Leftarrow DL[c \uplus p, b \cap q; c \sqcap \neg b](a) \Leftarrow p(a) \wedge \neg q(a)$.

We observe that the reason behind circular justifications is that the weak/strong answer set semantics does not satisfy the condition of well-supportedness. The notion of well-supportedness is introduced in [Fages, 1994]. For a normal logic program $R$, an interpretation $I$ is *well-supported* if there exists a strict well-founded partial order $\prec$ on $I$ such that for any $a \in I$, there is a rule $a \leftarrow body(r)$ in $ground(R)$ such that $I$ satisfies $body(r)$ and for every positive literal $b$ in $body(r)$, $b \prec a$. A binary relation $\leq$ is *well-founded* if there is no infinite decreasing chain $a_0 \geq a_1 \geq \cdots$. Clearly, a well-supported interpretation $I$ guarantees that every $a \in I$ is free of circular justifications.

Fages' definition of well-supportedness does not apply to DL-programs. On the one hand, positive literals in a normal logic program are only atoms, but in a DL-program, positive literals in rule bodies may be dl-atoms. On the other hand, circular justifications in a normal logic program are caused only by positive literals in rule bodies, but in a DL-program, some circular justifications are caused by negative literals.

In this section, we extend Fages' well-supportedness to DL-programs and define new answer set semantics for DL-programs that satisfy the extended well-supportedness condition. We start by introducing a notion of *up to satisfaction*.

**Definition 2** Let $KB = (L, R)$ be a DL-program, and $E$ and $I$ be two sets of atoms in $HB_R$ with $E \subseteq I$. For any ground literal $l$, we define "$E$ up to $I$ satisfies $l$ under $L$," denoted $(E, I) \models_L l$, as follows:

1. For a ground atom $a \in HB_R$, $(E, I) \models_L a$ if $a \in E$; $(E, I) \models_L not\ a$ if $a \notin I$.
2. For a ground dl-atom $A$, $(E, I) \models_L A$ if for every $F$ with $E \subseteq F \subseteq I$, $F \models_L A$; $(E, I) \models_L not\ A$ if for no $F$ with $E \subseteq F \subseteq I$, $F \models_L A$.

As the phrase "up to" suggests, for any ground (positive or negative) literal $l$, $(E, I) \models_L l$ means that for all interpretations $F$ between $E$ and $I$, $F \models_L l$. This implies that the truth of $l$ depends on $E$ and $\neg I^-$ and is independent of atoms in $I \setminus E$, since for any $a \in I \setminus E$ and any interpretation $F$ with $E \subseteq F \subseteq I$, whether or not $a$ is in $F$, $F \models_L l$.

This up to satisfaction has the property of monotonicity.

**Proposition 1** *Let $A$ be a ground atom or ground dl-atom. For any $E_1 \subseteq E_2 \subseteq I$, if $(E_1, I) \models_L A$ then $(E_2, I) \models_L A$, and if $(E_1, I) \models_L not\ A$ then $(E_2, I) \models_L not\ A$.*

### 3.1 Well-Supportedness for DL-Programs

In normal logic programs, circular justifications are caused only by some positive literals in rule bodies. Therefore, in Fages' definition of well-supportedness, it is required that for any $a$ in an interpretation $I$, there should be a rule $a \leftarrow body(r)$ such that $I$ satisfies $body(r)$ and for every positive literal $b$ in $body(r)$, $b \prec a$.

Following Fages' definition, we introduce a form of well-supportedness for DL-programs, which overcomes circular justifications caused only by positive literals in rule bodies. The idea is as follows: for each $a \in I$ we require that there exist a rule $a \leftarrow body(r)$ and a proper subset $E$ of $I$ such that for each positive literal $l$ in $body(r)$, $(E, I) \models_L l$, where $E$ contains only atoms whose truth in $I$ is not circularly dependent on $a$. Recall that when $(E, I) \models_L l$, the truth of $l$ depends only on $E$ and $\neg I^-$, independently of $I \setminus E$. Therefore, the truth of $a \in I$ depends only on $E$ and $\neg I^-$ and thus $a$ will have no circular justification caused by positive literals in rule bodies. Formally, we have

**Definition 3** Let $KB = (L, R)$ be a DL-program. An interpretation $I$ is *weakly well-supported* if there exists a strict well-founded partial order $\prec$ on $I$ such that for any $a \in I$,

there is a rule $a \leftarrow body(r)$ in $ground(R)$ and a subset $E \subset I$ such that (1) $I \models_L body(r)$ and for every $b \in E$, $b \prec a$, and (2) for every positive literal $l$ in $body(r)$, $(E, I) \models_L l$.

The condition "for every $b \in E$, $b \prec a$" means that the truth of all atoms in $E$ is not circularly dependent on $a$. The following result shows that Definition 3 is a proper extension to Fages' well-supportedness.

**Theorem 1** *Let $KB = (L, R)$ be a DL-program, where $L = \emptyset$ and $R$ is a normal logic program. An interpretation $I$ is weakly well-supported iff $I$ is well-supported under Fages' definition.*

The following example illustrates that neither the weak nor the strong answer set semantics satisfies this extended well-supportedness condition, which explains why their answer sets incur circular justifications.

**Example 2** Consider $KB = (L, R)$ in Example 1. $I_1 = \emptyset$ and $I_2 = \{p(a)\}$ are both strong and weak answer sets of $KB$. Obviously, $I_1 = \emptyset$ is weakly well-supported. Consider $I_2 = \{p(a)\}$. For $p(a) \in I_2$, we have a single rule $r$ in $ground(R)$ with the head $p(a)$ and a single proper subset $E = \emptyset$ of $I_2$. $body(r)$ has a single positive literal $l = DL[c \uplus p, b \cap q; c \sqcap \neg b](a)$. $I_2 \models_L body(r)$, but $(E, I_2) \not\models_L l$. Thus, $I_2$ is not weakly well-supported.

Weakly well-supported interpretations are free of circular justifications caused by positive literals in rule bodies. However, in DL-programs circular justifications may also be caused by some negative literals in rule bodies, as illustrated in the next example.

**Example 3** Let $KB = (L, R)$, where $L = \emptyset$ and $R = \{p(a) \leftarrow not\ DL[c \uplus p, c \cap p; \neg c](a)\}$. $I_1 = \emptyset$ and $I_2 = \{p(a)\}$ are both strong and weak answer sets of $KB$. They are also weakly well-supported. For $p(a) \in I_2$, however, we have a circular justification, $p(a) \Leftarrow not\ DL[c \uplus p, c \cap p; \neg c](a) \Leftarrow p(a)$, which is caused by the negative literal in the rule body.

To eliminate all circular justifications, we introduce another form of well-supportedness.

**Definition 4** Let $KB = (L, R)$ be a DL-program. An interpretation $I$ is *strongly well-supported* if there exists a strict well-founded partial order $\prec$ on $I$ such that for any $a \in I$, there is a rule $a \leftarrow body(r)$ in $ground(R)$ and a subset $E \subset I$ such that $(E, I) \models_L body(r)$ and for every $b \in E$, $b \prec a$.

The conditions "$(E, I) \models_L body(r)$" and "for every $b \in E, b \prec a$" suggest that the truth of $a$ is determined by $body(r)$ whose truth is determined by $E$ and $\neg I^-$, where for no $b \in E$, its truth is circularly dependent on $a$. This eliminates circular justifications on $a$, caused either by positive literals or negative literals in rule bodies.

**Proposition 2** *If an interpretation $I$ is strongly well-supported, then $I$ is weakly well-supported.*

The converse does not hold. For instance, in Example 3, $I_2 = \{p(a)\}$ is weakly well-supported. For $p(a) \in I_2$, we have a single rule $r$ in $ground(R)$ with the head $p(a)$ and a single proper subset $E = \emptyset$ of $I_2$. Although $I_2 \models_L body(r)$, $(E, I_2) \not\models_L body(r)$. Thus $I_2$ is not strongly well-supported.

However, when a DL-program contains no negative dl-atoms in rule bodies, the two forms of well-supportedness coincide.

**Theorem 2** *For a DL-program $KB = (L, R)$, where $R$ contains no negative dl-atoms, an interpretation $I$ is strongly well-supported iff $I$ is weakly well-supported.*

Since normal logic programs can be viewed as special DL-programs without dl-atoms, the following corollary is immediate from Theorems 1 and 2.

**Corollary 1** *Let $KB = (L, R)$ be a DL-program, where $L = \emptyset$ and $R$ is a normal logic program. An interpretation $I$ is strongly well-supported iff $I$ is well-supported under Fages' definition.*

### 3.2 Well-Supported Semantics for DL-Programs

We define answer sets for DL-programs that are either weakly well-supported or strongly well-supported models. First, we introduce an immediate consequence operator $\mathcal{T}_{KB}$ in terms of the up to satisfaction $(E, I) \models_L l$.

**Definition 5** Let $KB = (L, R)$ be a DL-program and $E$ and $I$ be two sets of ground atoms with $E \subseteq I \subseteq HB_R$. Define

$$\mathcal{T}_{KB}(E, I) = \{\, a \mid a \leftarrow body(r) \in ground(R) \text{ and } (E, I) \models_L body(r) \}.$$

A key property is that when the second argument $I$ is a model of $KB$, $\mathcal{T}_{KB}$ is monotone w.r.t. its first argument $E$.

**Theorem 3** *Let $KB = (L, R)$ be a DL-program and $I$ a model of $KB$. For any $E_1 \subseteq E_2 \subseteq I$, $\mathcal{T}_{KB}(E_1, I) \subseteq \mathcal{T}_{KB}(E_2, I) \subseteq I$.*

Therefore, for any model $I$ of $KB$, the monotone sequence $\langle \mathcal{T}^i_{KB}(\emptyset, I) \rangle^{\infty}_{i=0}$, where $\mathcal{T}^0_{KB}(\emptyset, I) = \emptyset$ and $\mathcal{T}^{i+1}_{KB}(\emptyset, I) = \mathcal{T}_{KB}(\mathcal{T}^i_{KB}(\emptyset, I), I)$, converges to a fixpoint, denoted $\mathcal{T}^{\alpha}_{KB}(\emptyset, I)$. This fixpoint has the following important property.

**Theorem 4** *Let $KB = (L, R)$ be a DL-program and $I$ a model of $KB$. If $I = \mathcal{T}^{\alpha}_{KB}(\emptyset, I)$, then $I$ is a minimal model of $KB$.*

Next, we define two new answer set semantics for DL-programs in terms of $\mathcal{T}_{KB}$. The first one is based on an extended Gelfond-Lifschitz transformation. Recall that for a normal logic program $R$ and an interpretation $I$, the *standard Gelfond-Lifschitz transformation* of $R$ w.r.t. $I$ is

$$R^I = \{\, a \leftarrow pos(r) \mid a \leftarrow pos(r), neg(r) \in ground(R) \text{ and } I \text{ satisfies } neg(r) \}.$$

Since $R^I$ has no negative literals in rule bodies, it has the least model. The *standard ASP semantics* defines $I$ to be an answer set of $R$ if $I$ is the least model of $R^I$ [Gelfond and Lifschitz, 1991].

We extend the standard Gelfond-Lifschitz transformation to DL-programs simply by replacing the satisfaction relation for normal logic programs with the satisfaction relation $\models_L$ for DL-programs, as formulated below.

**Definition 6** Given a DL-program $KB = (L, R)$ and an interpretation $I$, the *Gelfond-Lifschitz transformation* of $R$ w.r.t. $I$ *under* $L$ is

$$R^I = \{\ a \leftarrow pos(r) \mid a \leftarrow pos(r), neg(r) \in ground(R) \text{ and } I \models_L neg(r)\}.$$

We then apply this extended Gelfond-Lifschitz transformation to define answer sets for DL-programs.

**Definition 7** Let $KB = (L, R)$ be a DL-program, $I$ a model of $KB$, and $KB^I = (L, R^I)$. $I$ is an *answer set* of $KB$ if $I$ coincides with the fixpoint $\mathcal{T}^{\alpha}_{KB^I}(\emptyset, I)$.

**Example 4** Consider $KB = (L, R)$ of Example 1. For $I_1 = \emptyset$, the Gelfond-Lifschitz transformation of $R$ w.r.t. $I$ under $L$ is $R^{I_1} = \emptyset$, so the fixpoint $\mathcal{T}^{\alpha}_{KB^{I_1}}(\emptyset, I_1)$ is empty. Therefore, $I_1$ is an answer set of $KB$. For $I_2 = \{p(a)\}$, the extended Gelfond-Lifschitz transformation $R^{I_2} = R$ and the fixpoint $\mathcal{T}^{\alpha}_{KB^{I_2}}(\emptyset, I_2) = \emptyset$. Since $I_2 \neq \mathcal{T}^{\alpha}_{KB^{I_2}}(\emptyset, I_2)$, $I_2$ is not an answer set of $KB$.

Such answer sets can be characterized by weakly well-supported models.

**Theorem 5** *Let $KB = (L, R)$ be a DL-program and $I$ a model of $KB$. $I$ is an answer set of $KB$ under Definition 7 iff $I$ is a weakly well-supported model of $KB$.*

Due to this result, we call answer sets under Definition 7 *weakly well-supported answer sets*.

**Theorem 6** *For any DL-program $KB$, weakly well-supported answer sets of $KB$ are strong answer sets of $KB$.*

As Example 4 shows, strong answer sets may not be weakly well-supported answer sets.

As illustrated in Example 3, weakly well-supported models may have circular justifications caused by negative dl-atoms in rule bodies. Such circular justifications persist in weakly well-supported answer sets. Observe that in the extended Gelfond-Lifschitz transformation $R^I$, all negative literals in rule bodies are dropped so that circular justifications caused by negative dl-atoms are completely missed. To overcome this problem, we present another definition of answer sets without using the extended Gelfond-Lifschitz transformation.

**Definition 8** Let $KB = (L, R)$ be a DL-program. A model $I$ of $KB$ is an *answer set* of $KB$ if $I$ coincides with the fixpoint $\mathcal{T}^{\alpha}_{KB}(\emptyset, I)$.

**Example 5** Consider $KB = (L, R)$ in Example 3, where $R = \{p(a) \leftarrow not\ DL[c \uplus p, c \cap p; \neg c](a)\}$. $I_1 = \emptyset$ and $I_2 = \{p(a)\}$ are models of $KB$. $\mathcal{T}^{\alpha}_{KB}(\emptyset, I_1) = \emptyset$, so $I_1$ is an answer set. For $I_2$, $\mathcal{T}^{0}_{KB}(\emptyset, I_2) = \emptyset$ and $\mathcal{T}^{1}_{KB}(\emptyset, I_2) = \mathcal{T}_{KB}(\emptyset, I_2) = \emptyset$, so $\mathcal{T}^{\alpha}_{KB}(\emptyset, I_2) = \emptyset$. Therefore, $I_2$ is not an answer set of $KB$.

Answer sets under Definition 8 can be characterized by strongly well-supported models.

**Theorem 7** *Let $KB = (L, R)$ be a DL-program and $I$ a model of $KB$. $I$ is an answer set of $KB$ under Definition 8 iff $I$ is a strongly well-supported model of $KB$.*

Due to this property, we call such answer sets *strongly well-supported answer sets*. The following corollary is immediate from Theorem 4.

**Corollary 2** *For any DL-program $KB$, strongly well-supported answer sets of $KB$ are minimal models of $KB$.*

As shown in Example 3, weakly well-supported (resp. strong or weak) answer sets are not necessarily minimal models. The following result follows from Proposition 2.

**Corollary 3** *For any DL-program $KB$, strongly well-supported answer sets of $KB$ are also weakly well-supported answer sets of $KB$.*

The next result follows from Theorem 2.

**Corollary 4** *Let $KB = (L, R)$ be a DL-program, where $R$ contains no rules with negative dl-atoms. A model $I$ of $KB$ is a strongly well-supported answer set iff $I$ is a weakly well-supported answer set.*

As a result, for any DL-program $KB = (L, R)$, its strongly well-supported answer sets are also weakly well-supported answer sets that are also strong answer sets that are also weak answer sets. When $L = \emptyset$ and $R$ is a normal logic program, all of the four answer set semantics coincide with the standard ASP semantics.

## 4 Related Work

To deal with the circular justification problem with the weak answer set semantics, [Eiter *et al.*, 2008b] introduce the strong answer set semantics. However, examples show that this problem persists in the strong answer set semantics. In [Eiter *et al.*, 2005], the authors also propose another answer set semantics for DL-programs by means of FLP-reduct [Faber *et al.*, 2004]. Given a DL-program $KB = (L, R)$ and an interpretation $I$, the *FLP-reduct* of $R$ w.r.t. $I$, denoted $fR^I$, is the set of all rules $r \in ground(R)$ such that $I \models_L body(r)$. Then, $I$ is defined to be an answer set of $KB$ if $I$ is a minimal model of $fR^I$.

FLP-reduct based answer sets of $KB$ are minimal models of $KB$. This shows an advantage over the weak/strong answer set semantics. However, it turns out that the problem of circular justifications occurring in weak/strong answer sets persists in FLP-reduct based answer sets. As an illustration, consider a DL-program $KB = (L, R)$, where $L = \emptyset$ and $R$ consists of two rules: $p(a) \leftarrow q(a)$ and $q(a) \leftarrow DL[c \uplus p, b \cap q; c \sqcup \neg b](a)$. $KB$ has only one model $I = \{p(a), q(a)\}$. The FLP-reduct $fR^I$ of $R$ w.r.t. $I$ is the same as $R$. Therefore, $I$ is an answer set of $KB$ under the FLP-reduct based semantics. We see that $p(a) \in I$ is circularly justified by a self-supporting loop: $p(a) \Leftarrow q(a) \Leftarrow DL[c \uplus p, b \cap q; c \sqcup \neg b](a) \Leftarrow p(a) \vee \neg q(a)$. Note that $p(a) \Leftarrow q(a) \Leftarrow \cdots \Leftarrow p(a) \vee \neg q(a)$ implies $p(a) \Leftarrow q(a) \Leftarrow \cdots \Leftarrow p(a)$, since the truth of $q(a)$ cannot be supported by $\neg q(a)$. It is easy to check that $I$ is neither strongly nor weakly well-supported.

The notion of FLP-reduct is first introduced in [Faber *et al.*, 2004], where it is used to define an answer set semantics for logic programs with aggregates (or abstract constraints). Our definition of the up to satisfaction relation $(E, I) \models_L l$ (Definition 2) is inspired by the notion of *conditional satisfaction* introduced by [Son *et al.*, 2007], which is also used to define an answer set semantics for logic programs with

abstract constraints. Although logic programs with abstract constraints and DL-programs are significantly different, their semantical issues are closely related. This is first observed by [Eiter *et al.*, 2008b].

To address the problem of circular justifications, recently [Wang *et al.*, 2010] extend the notion of *loop formulas* [Lin and Zhao, 2004] to DL-programs and propose a *canonical answer set semantics* by means of loop formulas. This approach first constructs the completion COMP($KB$) of a DL-program $KB$, then constructs the set LF($KB$) of all loop formulas from $KB$, and finally defines a *canonical answer set* to be a model of COMP($KB$) $\cup$ LF($KB$). It turns out that this loop formula based semantics also incurs circular justifications. Consider a DL-program $KB = (L, R)$, where $L = \{\neg e(b), f(a)\}$ and $R$ consists of two rules: $p(b) \leftarrow p(a)$ and $p(a) \leftarrow DL[e \uplus p, f \cap p; f](b)$. $I = \{p(a), p(b)\}$ is a canonical answer set of $KB$, but $p(b) \in I$ is circularly justified by a self-supporting loop: $p(b) \Leftarrow p(a) \Leftarrow DL[e \uplus p, f \cap p; f](b) \Leftarrow \neg p(a) \vee p(b)$. It is easy to check that $I$ is neither strongly nor weakly well-supported.

## 5 Summary

[Fages, 1994] introduces the notion of well-supportedness as a key requirement for the semantics of normal logic programs and characterizes the standard ASP semantics in terms of the well-supportedness condition. In this paper, we extend Fages' work to DL-programs. We introduce two forms of well-supportedness for DL-programs. The first one defines weakly well-supported models that are free of circular justifications caused by positive literals in rule bodies. The second one defines strongly well-supported models that are free of circular justifications caused by either positive or negative literals. We then define two new answer set semantics for DL-programs and characterize them in terms of weakly and strongly well-supported models, respectively. While the weakly well-supported answer set semantics is free of circular justifications for the class of DL-programs without negative dl-atoms, the strongly well-supported semantics is free of circular justifications for all DL-programs. Our simple examples demonstrate that the existing answer set semantics for DL-programs, such as the weak answer set semantics, the strong answer set semantics, and the FLP-based answer set semantics, satisfy neither the strong nor the weak well-supportedness condition, even for DL-programs without negative dl-atoms. This explains why their answer sets incur circular justifications.

As interesting future work, we are extending the well-supportedness to disjunctive DL-programs, where the head of a rule is a disjunction of atoms. We are also deeply exploiting the semantical connections between DL-programs and logic programs with abstract constraints. Moreover, practically implementing the proposed well-supported semantics for DL-programs presents a challenging task.

## Acknowledgments

We would like to thank all anonymous reviewers for their helpful comments. This work is supported in part by NSFC grants 60970045 and 60833001.

## References

[Baader *et al.*, 2003] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.

[de Bruijn *et al.*, 2008] J. de Bruijn, T. Eiter, and H. Tompits. Embedding approaches to combining rules and ontologies into autoepistemic logic. In *KR*, pages 485–495, 2008.

[Eiter *et al.*, 2005] T. Eiter, G. Ianni, R. Schindlauer, and H. Tompits. A uniform integration of higher-order reasoning and external evaluations in answer-set programming. In *IJCAI*, pages 90–96, 2005.

[Eiter *et al.*, 2008a] T. Eiter, G. Ianni, T. Krennwallner, and A. Polleres. Rules and ontologies for the semantic web. In *Proceedings of the 2nd International Conference on Web Reasoning and Rule Systems (RR-08)*, pages 1–53, 2008.

[Eiter *et al.*, 2008b] T. 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.

[Faber *et al.*, 2004] W. Faber, N. Leone, and G. Pfeifer. Recursive aggregates in disjunctive logic programs: Semantics and complexity. In *Logics in Artificial Intelligence: European Workshop (JELIA-04)*, pages 200–212, 2004.

[Fages, 1994] François Fages. Consistency of clark's completion and existence of stable models. *Journal of Methods of Logic in Computer Science*, 1:51–60, 1994.

[Gelfond and Lifschitz, 1991] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[Horrocks *et al.*, 2003] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From SHIQ and RDF to OWL: the making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.

[Lin and Zhao, 2004] F. Lin and Y. Zhao. Assat: computing answer sets of a logic program by sat solvers. *Artificial Intelligence*, 157(1-2):115–137, 2004.

[Lukasiewicz, 2010] T. Lukasiewicz. A novel combination of answer set programming with description logics for the semantic web. *IEEE Transactions on Knowledge and Data Engineering*, 22(11):1577–1592, 2010.

[Motik and Rosati, 2010] B. Motik and R. Rosati. Reconciling description logics and rules. *J. ACM*, 57(5), 2010.

[Rosati, 2006] R. Rosati. DL+log: Tight integration of description logics and disjunctive datalog. In *KR-06*, pages 68–78, 2006.

[Son *et al.*, 2007] T. C. Son, E. Pontelli, and P. H. Tu. Answer sets for logic programs with arbitrary abstract constraint atoms. *Journal of Artificial Intelligence Research*, 29:353–389, 2007.

[Wang *et al.*, 2010] Y. Wang, J. H. You, L. Y. Yuan, and Y. D. Shen. Loop formulas for description logic programs. *TPLP*, 10(4-6):531–545, 2010.