RECONSIDERING ANSWER SET SEMANTICS FOR DISJUNCTIVE LOGIC PROGRAMS

YI-DONG SHEN  THOMAS EITER

LOGCOMP Research Report 18-03
FEBRUARY 2018
Abstract. Gelfond and Lifschitz (NGC, 1991) presented the seminal answer set semantics (i.e., the GL-semantics) for simple disjunctive programs, which has been taken as the standard answer set semantics in the literature. However, our recent observation shows that the GL-semantics has limitations because its unique rule head atom requirement is a bit too restrictive and may exclude some desired answer sets. In this paper we reconsider answer set semantics for disjunctive programs and present a novel alternative, called the determining inference semantics (DI-semantics for short), which overcomes the limitations of the GL-semantics. In particular, we introduce a head formula selection function $sel$, which is an analog of the Hilbert’s epsilon operator $\epsilon$ adapted to the context of answer set programming, to formalize the nondeterministic inference operator $|\cdot|$; propose a disjunctive program reduct $\Pi^I_{sel}$ w.r.t. a selection function $sel$ and an interpretation $I$ to transform a disjunctive program $\Pi$ into a normal program, apply to $\Pi^I_{sel}$ an answer set semantics $X$ for normal programs to compute candidate answer sets of $\Pi$, and finally define $I$ to be a DI-answer set of $\Pi$ if $I$ is minimal among all candidate answer sets. The DI-semantics is general and applicable to extend any answer set semantics $X$ for normal programs to disjunctive programs. As two particularly interesting application scenarios, we use the DI-semantics to formalize a Generalized Strategic Companies problem and a relaxed version of the well-known Russell’s Barber paradox. Finally we give computational complexity results; in particular we show that in the propositional case deciding whether a simple disjunctive program $\Pi$ has DI-answer sets is NP-complete; this is an additional advantage over the GL-semantics, as deciding whether $\Pi$ has GL-answer sets is $\Sigma^p_2$-complete.
Contents

1 Introduction 3

2 Preliminaries 7
   2.1 A first-order logic language ....................................................... 7
   2.2 General disjunctive programs ...................................................... 7
       2.2.1 Grounding ......................................................................... 8
       2.2.2 Satisfaction and models ....................................................... 8
   2.3 GL-semantics for simple disjunctive/normal programs ......................... 8

3 A General Answer Set Semantics for Disjunctive Programs 9

4 Determining Inference Semantics for Simple Disjunctive Programs 10

5 Determining Inference Semantics for General Disjunctive Programs 15

6 Computational Complexity 19
   6.1 DI-answer sets for other base semantics ......................................... 22

7 Related Work 22
   7.1 FLP-semantics ........................................................................ 23
   7.2 Equilibrium logic-based semantics .............................................. 24
   7.3 Split programs ....................................................................... 24

8 Conclusion 26
1 Introduction

Answer set programming (ASP) is a major declarative programming paradigm in knowledge representation and reasoning (Marek and Truszczyński, 1999; Niemela, 1999), which is specifically oriented towards modeling and solving combinatorial search problems arising in many application areas of AI such as planning, reasoning about actions, diagnosis, and beyond (Lifschitz, 2002; Baral, 2003; Brewka et al., 2011, 2016). The idea of ASP is to represent a problem by a logic program whose answer sets correspond to solutions. Generally speaking, in syntax logic programs in ASP could be viewed as a first-order logic language extended with inference operators such as the if-then rule operator \( \leftarrow \) together with the disjunctive rule head operator \( | \); regarding semantics, the meaning of a logic program under ASP is given by a set of intended models, called stable models or answer sets, which correspond to solutions to the given problem (Gelfond and Lifschitz, 1988, 1991).

In the very beginning of the ASP evolution, only simple normal programs were considered (Gelfond and Lifschitz, 1988), which consist of rules of the form

\[
A \leftarrow B_1 \land \cdots \land B_m \land \neg C_1 \land \cdots \land \neg C_n
\]

(1)

where \( A \) and every \( B_i \) and \( C_i \) are atoms. \( \leftarrow \) is an if-then inference operator such that by applying rule (1) we infer the rule head \( A \) if the rule body \( B_1 \land \cdots \land B_m \land \neg C_1 \land \cdots \land \neg C_n \) is satisfied.

In order to support nondeterministic inferences, i.e., to nondeterministically infer an atom among several atoms when some conditions hold, Gelfond and Lifschitz (1991) further introduced simple disjunctive programs consisting of rules with disjunctive heads of the form

\[
A_1 | \cdots | A_k \leftarrow B_1 \land \cdots \land B_m \land \neg C_1 \land \cdots \land \neg C_n
\]

(2)

where every \( A_i, B_i \) and \( C_i \) are atoms. \( | \) in combination with \( \leftarrow \) is an inference operator such that by applying rule (2) we nondeterministically infer some \( A_i \) among \( A_1, \cdots, A_k \) if \( B_1 \land \cdots \land B_m \land \neg C_1 \land \cdots \land \neg C_n \) is satisfied. These rules were further extended to more general forms containing aggregates (Faber et al., 2004; Pelov, 2004; Shen and You, 2007; Son et al., 2007; Faber et al., 2011; Asuncion et al., 2015), external source atoms (Eiter et al., 2005, 2008), propositional formulas (Pearce, 1996; Ferraris, 2005; Truszczynski, 2010), or first-order formulas (Bartholomew et al., 2011; Ferraris et al., 2011; Shen et al., 2014).\(^1\)

Gelfond and Lifschitz (1991) defined the seminal answer set semantics for a simple disjunctive program \( \Pi \), i.e., an interpretation \( I \) is an answer set of \( \Pi \) if \( I \) is a minimal model of the GL-reduct \( \Pi^I \) of \( \Pi \), where \( \Pi^I \) is obtained by first removing all rules (2) containing a \( C_i \in I \) and then removing all \( \neg C_i \) from the remaining rules. This GL-semantics has been widely accepted in the ASP community and all semantics of extended disjunctive programs presented in the literature are supposed to agree with it (Faber et al., 2011; Ferraris, 2005; Pearce, 1996; Truszczynski, 2010; Asuncion et al., 2015); thus it is often called the standard answer set semantics for simple disjunctive programs.

The GL-semantics is intuitively justified on the basis of the rationality principle (Gelfond, 2008), i.e., one shall not believe anything one is not forced to believe; specifically, Gelfond (2008) argued that we are forced to believe all atoms in an answer set of the GL-semantics because as shown by Baral and Gelfond (1994), they have the following critical property:

\(^1\)Rules in a disjunctive program could also contain strong negations of the form \( \neg A \), where \( A \) is an atom. However, as shown in (Gelfond and Lifschitz, 1991), for any atom \( p(X) \) the strong negation \( \neg p(X) \) can be compiled away by introducing a fresh predicate \( p'(X) \), i.e., \( \neg p(X) \) can be replaced by \( p'(X) \) along with a rule \( \bot \leftarrow p(X) \land p'(X) \).
Unique rule head atom property of the GL-semantics: Let $I$ be an answer set of a simple disjunctive program $\Pi$ under the GL-semantics. Then, for every $A$ in $I$ there is a rule (2) in $\Pi$ such that $I$ satisfies the rule body and $A$ is the only atom in the rule head contained in $I$.

For instance, let $\Pi = \{p | q\}$; then $I = \{p, q\}$ could not be an answer set under the GL-semantics because it violates the unique rule head atom property. Another typical example is that $I = \{p\}$ could not be an answer set of $\Pi = \{p \leftarrow \neg p\}$.

In this paper we reconsider answer set semantics for disjunctive programs and start by raising a fundamental question: is the GL-semantics the only answer set semantics for simple disjunctive programs? Specifically, we question the appropriateness of taking the unique rule head atom property as a necessary condition of an answer set.

Our recent observation shows that the GL-semantics has limitations because its unique rule head atom requirement is a bit too restrictive and may exclude some desired answer sets. The following real life example well illustrates the limitations.

Example 1 Consider the following simple disjunctive program for a Chinese-English joint PhD program:

\[
\Pi : \quad \text{thesisInChinese}(X) \mid \text{thesisInEnglish}(X) \leftarrow \text{student}(X)
\]

\[
\text{student}(\text{haoXue}) \quad r_1
\]

\[
\text{majorInEnglish}(\text{haoXue}) \quad r_2
\]

\[
\text{thesisInChinese}(\text{haoXue}) \quad r_3
\]

\[
f \leftarrow \text{majorInEnglish}(X) \land \neg \text{thesisInEnglish}(X) \land \neg f \quad r_4
\]

Rule $r_1$ says that any PhD thesis is written either in Chinese or English, while $r_5$ further requires that one who majors in English should have a thesis written in English. Then, given the facts that $\text{haoXue}$ is a PhD student ($r_2$) majoring in English ($r_3$) and has written a thesis in Chinese ($r_4$), what are the solutions (i.e., answer sets) to the problem expressed by $\Pi$?

Intuitively, by applying the two rules $r_1$ and $r_2$ we infer one of $\text{thesisInChinese}(\text{haoXue})$ and $\text{thesisInEnglish}(\text{haoXue})$, nondeterministically. Thus, by applying the three rules $r_1, r_2, r_4$ we either infer

\[
\text{thesisInChinese}(\text{haoXue})
\]

(from $r_1, r_2$ as well as from $r_4$) or

\[
\text{thesisInEnglish}(\text{haoXue}), \text{thesisInChinese}(\text{haoXue})
\]

(from $r_1, r_2$ and from $r_4$, respectively). Then, by applying the four rules $r_1\cdots r_4$ we obtain two potential answer sets for $\Pi$:

\[
I_1 = \{\text{student}(\text{haoXue}), \text{majorInEnglish}(\text{haoXue}), \text{thesisInChinese}(\text{haoXue})\},
\]

\[
I_2 = \{\text{student}(\text{haoXue}), \text{majorInEnglish}(\text{haoXue}), \text{thesisInChinese}(\text{haoXue}), \text{thesisInEnglish}(\text{haoXue})\}.
\]

Rule $r_5$ is a constraint stating that for any individual $X$ no answer set satisfies the condition

\[
\text{majorInEnglish}(X) \land \neg \text{thesisInEnglish}(X);
\]

this excludes $I_1$ which contains $\text{majorInEnglish}(\text{haoXue})$ but not $\text{thesisInEnglish}(\text{haoXue})$. As a result, by applying the five rules we infer $I_2$ as the only candidate answer set for $\Pi$. As $I_2$ is minimal in the sense...
that no \( J \subset I_2 \) is a candidate answer set for \( \Pi \), we expect \( I_2 \) to be an answer set of \( \Pi \). The solution \( I_2 \) tells us that the PhD student haoXue has prepared two versions for his thesis, one in Chinese and the other in English.

However, contrary to our expectation, this disjunctive program has no answer set under the GL-semantics: \( I_2 \) is not an answer set under the GL-semantics because it violates the unique rule head atom property, where for \( \text{thesisInEnglish}(\text{haoXue}) \in I_2 \) there is no rule in \( \Pi \) such that \( I_2 \) satisfies the rule body and \( \text{thesisInEnglish}(\text{haoXue}) \) is the only atom in the rule head contained in \( I_2 \).

Under the unique rule head atom property, given a rule

\[
A_1 | \cdots | A_k
\]

any other rule with a head

\[
A_1 | \cdots | A_k | A_{k+1} \cdots | A_l \quad (l > k)
\]
is redundant and thus can be deleted because no answer set under the GL-semantics will contain some \( A_i \) with \( k < i < l \). This amounts to treating the inference operator \( \vert \) as the logical connective \( \lor \) for classical disjunction, where \( (A_1 \lor \cdots \lor A_k) \land (A_1 \lor \cdots \lor A_k \lor A_{k+1} \cdots \lor A_l) \) is equivalent to \( A_1 \lor \cdots \lor A_k \). Obviously, this does not comply with our intuition that by applying the above two rules we nondeterministically infer some \( A_i, 1 \leq i \leq k \), from the first head, and some \( A_j, 1 \leq j \leq l \), from the second. It is such nondeterministic inferences via disjunctive rule heads that allow us to generate different candidate answer sets. In this sense, we may well refer to \( \vert \) as a nondeterministic inference operator.

As an inference operator, \( \vert \) essentially differs from the logical disjunction connective \( \lor \). This is analogous to the if-then inference operator \( \leftarrow \) which essentially differs from the logical implication connective \( \supset \). For example, by \( p \lor \neg p \) in a logical formula we mean a logical tautology, while by \( p \vert \neg p \) in a rule head we mean to nondeterministically infer \( p \) or \( \neg p \). Our careful study reveals that the difference between \( \lor \) and \( \vert \) is akin to the difference between the existential quantifier \( \exists \) and Hilbert’s epsilon operator \( \varepsilon \). Let \( D = \{ t_1, \ldots, t_m \} \) be the domain of a variable \( x \). Hilbert redefines \( (\exists x)F(x) \) as \( F(\varepsilon x \ F) \), where \( \varepsilon x \ F \) returns some term \( t \) in the domain \( D \) such that \( F(t) \) is true, otherwise it returns some default or arbitrary term. If there is more than one term making \( F \) true, then any one of these terms can be chosen, nondeterministically. As a result, \( (\exists x)F(x) \) amounts to \( F(t_1) \lor \cdots \lor F(t_m) \), while \( F(\varepsilon x \ F) \) nondeterministically selects one \( F(t) \) among \( F(t_1), \ldots, F(t_m) \) and thus amounts to \( F(t_1) \lor \cdots \lor F(t_m) \).

The following real world example further illustrates the necessity to use the operator \( \vert \) and the difference between \( \land \) and \( \lor \).

**Example 2** Consider the following more general disjunctive program for animal classification:

\[
\Pi : \text{mammal}(X) \supset \text{vertebrate}(X) \quad \text{vertebrate}(X) \supset \text{mammal}(X) \quad \text{animal}(\text{grus}_\text{grus}) \\
\text{animal(} \text{grus}_\text{grus} \text{)} \quad \text{mammal(} \text{grus}_\text{grus} \text{)} \\
\bot \leftarrow \neg \text{vertebrate}(\text{grus}_\text{grus}) \\
\]

Rule \( r_1 \) expresses a class subsumption relation with uncertainty, i.e., one does not know whether \( \text{vertebrate} \) is subsumed by \( \text{mammal} \) or the other way around. Note that we cannot rewrite \( r_1 \) as \( r'_1 \) by replacing the inference operator \( \vert \) with the logical connective \( \lor \), as the head of \( r'_1 \) is simply a tautology. Rule \( r_4 \) says that it is not the case that \( \text{grus}_\text{grus} \) is not \( \text{vertebrate} \). Intuitively, by applying the two rules \( r_1, r_2 \) we infer \( \text{mammal(} \text{grus}_\text{grus} \text{)} \supset \text{vertebrate(} \text{grus}_\text{grus} \text{)} \) or \( \text{vertebrate(} \text{grus}_\text{grus} \text{)} \supset \text{mammal(} \text{grus}_\text{grus} \text{)} \),
nondeterministically. Then, by choosing the former we infer from the three rules \( r_1, r_2, r_3 \) a potential answer set

\[
I = \{ \text{animal(grus_grus)}, \text{mammal(grus_grus)}, \text{vertebrate(grus_grus)} \}
\]

(i.e., \( \text{mammal(grus_grus)} \sqsupset \text{vertebrate(grus_grus)} \)) together with \( \text{mammal(grus_grus)} \) logically entails \( \text{vertebrate(grus_grus)} \). \( I \) satisfies the constraint \( r_4 \) and thus is a candidate answer set for \( \Pi \). As \( I \) is a minimal model of \( \Pi \), no model \( J \subset I \) of \( \Pi \) would be another candidate answer set; hence we expect \( I \) to be an answer set of \( \Pi \).

However, contrary to our expectation, \( \Pi \) has no answer set under all of the existing answer set semantics for disjunctive programs such as those defined in (Pearce, 2006; Truszczynski, 2010; Bartholomew et al., 2011; Faber et al., 2011; Ferraris et al., 2011).

The above significant observations motivate us to explore new answer set semantics for disjunctive programs. The main contributions of this paper are summarized as follows.

(1) We present a general answer set semantics for disjunctive programs, called determining inference semantics (DI-semantics for short). The basic idea is as follows: (i) We introduce a selection function \( \text{sel} \) to formalize the nondeterministic inference operator \( \mid \); i.e., for every interpretation \( I \) and every rule head \( H_1 | \cdots | H_k, \text{sel}(H_1 | \cdots | H_k, I) \) nondeterministically selects one \( H_i \) satisfied by \( I \). We may view \( \text{sel} \) as an analog of the Hilbert’s epsilon operator \( \epsilon \) for the ASP scenario. (ii) Given an interpretation \( I \) and a selection function \( \text{sel} \), we transform a disjunctive program \( \Pi \) into a normal program \( \Pi'_{\text{sel}} \) called a disjunctive program reduct, such that for every rule \( \text{head}(r) \leftarrow \text{body}(r) \) in \( \Pi \), \( \text{sel}(\text{head}(r), I) \leftarrow \text{body}(r) \) is in \( \Pi'_{\text{sel}} \) if \( I \) satisfies \( \text{body}(r) \). (iii) We apply to \( \Pi'_{\text{sel}} \) any answer set semantics \( \mathcal{X} \) for normal programs to compute candidate answer sets of \( \Pi \); i.e., \( I \) is a candidate answer set of \( \Pi \) if \( I \) is an answer set of \( \Pi'_{\text{sel}} \) under the base semantics \( \mathcal{X} \). (iv) Finally, a model \( I \) of \( \Pi \) is a DI-answer set if \( I \) is minimal among all candidate answer sets.

(2) By replacing the base semantics \( \mathcal{X} \) in the above general semantics with the GL-semantics defined by Gelfond and Lifschitz (1988), we induce a DI-semantics for simple disjunctive programs. This new semantics does not have the unique rule head atom restriction and thus resolves the limitations of the GL-semantics. We show that an answer set under the GL-semantics is an answer set under the DI-semantics, but not vice versa. To clearly see the essential difference of the DI-semantics from the GL-semantics, we also present a new characterization of the GL-semantics in terms of a disjunctive program reduct \( \Pi'_{\text{sel}} \). It is based on this characterization that we obtain a satisfactory solution to the open problem presented by Hitzler and Seda (1999) about characterizing split normal derivatives of a simple disjunctive program \( \Pi \).

As two particularly interesting application scenarios, we propose to use the DI-semantics to formalize a Generalized Strategic Companies problem and a relaxed version of the well-known Russell’s Barber paradox.

(3) By replacing the base semantics \( \mathcal{X} \) in the above general semantics with the well-justified semantics defined by Shen et al. (2014), we further induce a DI-semantics for general disjunctive programs consisting of rules of the form \( H_1 | \cdots | H_k \leftarrow B \), where \( B \) and every \( H_i \) are arbitrary first-order formulas. This closes an important open issue presented in (Shen et al., 2014) about extending the well-justified semantics from general normal programs with rules of the form \( H_1 \leftarrow B \) to general disjunctive programs.

(4) Finally, we show that in the propositional case deciding whether a simple disjunctive program \( \Pi \) has some DI-answer set is \( \text{NP-complete} \), and deciding whether a ground literal is true in some (resp. every) DI-answer set of \( \Pi \) is \( \Sigma_2^p \)-complete (resp. \( \Pi_2^p \)-complete). This is an additional advantage over the GL-semantics,
as deciding whether a simple disjunctive program has GL-answer sets is \(\Sigma_2^p\)-complete (Eiter and Gottlob, 1995). For general disjunctive programs, the complexity increases by one level in the polynomial hierarchy.

2 Preliminaries

In this section, we first introduce a first-order logic language, and then define general disjunctive programs with first-order formulas.

2.1 A first-order logic language

We define a first-order logic language \(L_\Sigma\) with equality over a signature \(\Sigma = (P, F)\), where \(P\) and \(F\) are countable sets of predicate and function symbols, respectively; \(C \subseteq F\) denotes the set of 0-ary function symbols, which are called constants. Variables, terms, atoms and literals are defined as usual. We denote variables with strings starting with \(x, y, z, X, Y\) or \(Z\).

First-order formulas (briefly formulas) are constructed as usual from atoms using connectives \(\neg, \land, \lor, \top, \bot, \exists\) and \(\forall\), where \(\top\) and \(\bot\) are two 0-place logical connectives expressing true and false, respectively. Formulas are closed if they contain no free variables, i.e., each variable occurrence is in the scope of some quantifier. A first-order theory (or theory) is a set of closed formulas. Terms, atoms and formulas are ground if they have no variables. By \(N_\Sigma\) we denote the set of all ground terms of \(\Sigma\), and by \(H_\Sigma\) the set of all ground atoms.

We consider SNA interpretations, i.e., interpretations which employ the well-known standard names assumption (SNA) (de Bruijn et al., 2008; Motik and Rosati, 2010). An SNA interpretation (or interpretation for short) \(I\) of \(L_\Sigma\) is a subset of \(H_\Sigma\) such that for any ground atom \(A\), \(I\) satisfies \(A\) if \(A \in I\), and \(I\) satisfies \(\neg A\) if \(A \notin I\). The notion of satisfaction/models of a formula/theory in \(I\) is defined as usual. A theory \(T\) is consistent or satisfiable if \(T\) has a model. We say that \(T\) entails a closed formula \(F\), denoted \(T \models F\), if all models of \(T\) are models of \(F\). Furthermore \(F\) is true (resp. false) in an interpretation \(I\) if \(I\) satisfies (resp. does not satisfy) \(F\).

For an interpretation \(I\), we let \(I^- = H_\Sigma \setminus I\) and \(\neg I^- = \{ \neg A \mid A \in I^- \}\).

2.2 General disjunctive programs

We define logic programs by extending the above first-order logic language \(L_\Sigma\) to include the two inference rule operators \(\leftarrow\) and \(\mid\).

**Definition 1** A general disjunctive program (disjunctive program or logic program for short) is a finite set of rules of the form

\[
H_1 \mid \cdots \mid H_k \leftarrow B
\]

(3)

where \(k > 0\),\(^3\) and each \(H_i\) and \(B\) are first-order formulas.

For convenience, for a rule \(r\) of form (3) we refer to \(B\) and \(H_1 \mid \cdots \mid H_k\) as the body and head of \(r\), denoted body(\(r\)) and head(\(r\)), respectively. We also refer to each \(H_i\) as a head formula. When body(\(r\)) is empty, we drop the if-then rule operator \(\leftarrow\).

\(^2\)For convenience, we sometimes write \(G \supset H\) and \(H \subset G\) exchangeably.

\(^3\)When \(k = 0\), the rule can be rewritten as \(\bot \leftarrow B\).
Remark 1  Note that we define logic programs using formulas in classical logic as the basic building blocks, as e.g. in many nonmonotonic logics such as Default Logic (Reiter, 1980), Autoepistemic Logic (Moore, 1985), etc. As inference operators (instead of logical connectives), ← and | do not occur in any formulas in rule bodies and heads. Intuitively, rule (3) reads as follows: if the body $B$ is satisfied, then we nondeterministically infer one $H_i$ from the head. This makes a rule (3) essentially different from a logical formula $B \supset H_1 \lor \cdots \lor H_k$ which is equivalent to the formula $\neg B \lor H_1 \lor \cdots \lor H_k$, and from a rule $H_1 \lor \cdots \lor H_k \leftarrow B$ which reads as: if the body $B$ is satisfied, then we infer the formula $H_1 \lor \cdots \lor H_k$.

A general disjunctive program is a general normal program (normal program for short) if $k = 1$; a simple disjunctive program if each $H_i$ is an atom and $B$ is a conjunction of literals, and a simple normal program if additionally $k = 1$. A positive simple normal/disjunctive program is a simple normal/disjunctive program without negative literals.

The Herbrand universe of a logic program $\Pi$, denoted $HU_\Pi$, is the set of all ground terms constructed with constant and function symbols from $\Pi$, and the Herbrand base of $\Pi$, denoted $HB_\Pi$, is the set of all ground atoms constructed with terms and predicate symbols from $\Pi$. Any subset of $HB_\Pi$ is a Herbrand interpretation of $\Pi$.

Definition 2  A propositional program $\Pi$ is a logic program which contains no variables, no function symbols except constants, and no equalities.

2.2.1 Grounding

In a logic program $\Pi$, some rules may contain free variables. In ASP, these free variables will be instantiated by constants from a finite set, usually the set $C_\Pi$ of constants occurring in $\Pi$.

A closed instance of a rule is the rule with all free variables replaced by constants in $C_\Pi$. The grounding of $\Pi$, denoted $ground(\Pi)$, is the set of all closed instances of all rules in $\Pi$. Since $C_\Pi$ is finite, $ground(\Pi)$ is finite.

Note that each rule $r$ of form (3) with the set $S$ of free variables may also be viewed as a globally universally quantified rule $\forall S(r)$, where the domain of each variable in $S$ is $C_\Pi$ while the domain of the other (locally quantified) variables is $\mathcal{N}_S$. Only globally universally quantified variables will be instantiated over their domain $C_\Pi$ for the grounding $ground(\Pi)$.

To sum up, a logic program $\Pi$ is viewed as shorthand for $ground(\Pi)$, where each free variable in $\Pi$ is viewed as shorthand for constants in $C_\Pi$.

2.2.2 Satisfaction and models

We extend the satisfaction relation of $L_\Sigma$ to logic programs in the following way. An interpretation $I$ satisfies a rule head $H_1 \mid \cdots \mid H_k$ if it satisfies some $H_i$; $I$ satisfies a closed instance $r$ of a rule if it either satisfies $head(r)$ or it does not satisfy $body(r)$; $I$ is a model of a logic program $\Pi$ if $I$ satisfies every rule $r \in ground(\Pi)$. Moreover, a model $I$ of $\Pi$ is minimal if $\Pi$ has no model $J$ that is a proper subset of $I$.

2.3 GL-semantics for simple disjunctive/normal programs

Let $\Pi$ be a simple disjunctive program and $I$ an interpretation. The $GL$-reduct of $\Pi$ w.r.t. $I$, written as $\Pi^I$, is obtained from $ground(\Pi)$ by (1) removing all rules whose bodies contain some $\neg C_i$ with $C_i \in I$, and (2)
removing from the remaining rules all $\neg C_i$. Note that every rule in the GL-reduct $\Pi^f$ is of the form

$$A_1 \mid \cdots \mid A_k \leftarrow B_1 \land \cdots \land B_m$$  \hspace{1cm} (4)

where all $A_i$ and $B_i$ are ground atoms.

The *GL-semantics* defines $I$ to be an answer set of $\Pi$ (referred to as *GL-answer set*) if $I$ is a minimal model of $\Pi^f$ (Gelfond and Lifschitz, 1991). When $\Pi$ is a simple normal program, the *GL-nilp-semantics* defines $I$ to be an answer set of $\Pi$ if $I$ is the least model of $\Pi^f$ (Gelfond and Lifschitz, 1988). For simple normal programs, the GL-semantics reduces to the GL-nilp-semantics.

### 3 A General Answer Set Semantics for Disjunctive Programs

To resolve the limitations of the GL-semantics, in this section we present a new answer set semantics for disjunctive programs. The new semantics is *general* in the sense that it is applicable to extend any answer set semantics for normal programs to disjunctive programs. The basic idea is to transform a disjunctive program $\Pi$ into a normal program $\Pi^f_{sel}$ w.r.t. an interpretation $I$ as well as a rule head formula selection function $sel$, and apply to $\Pi^f_{sel}$ an answer set semantics for normal programs to compute candidate answer sets of $\Pi$. The rule head formula selection function formalizes the nondeterministic inference operator $|$ by nondeterministically selecting a head formula from every rule head.

**Definition 3 (head selection function for disjunctive programs)** Let $\Pi$ be a disjunctive program and $I$ the collection of all interpretations. Let $\mathcal{H}D_\Pi$ be the set of all rule heads in $\text{ground}(\Pi)$, and $\mathcal{H}F_\Pi$ the set of all head formulas in $\mathcal{H}D_\Pi$. A head selection for $\Pi$ is a function $sel : \mathcal{H}D_\Pi \times I \mapsto \mathcal{H}F_\Pi \cup \{\bot\}$ such that for every interpretation $I \in \mathcal{I}$ and every rule $r \in \text{ground}(\Pi)$,

$$sel(\text{head}(r), I) = \begin{cases} H_i, & \text{if some head formula } H_i \text{ in } \text{head}(r) \text{ is satisfied by } I \\ \bot, & \text{otherwise.} \end{cases}$$

For example, let $\Pi = \{p \mid q, q \mid s\}$ and $I = \{p, q\}$; then there are two head selection functions $sel_1$ and $sel_2$ for $\Pi$ on $I$, where $sel_1(p \mid q, I) = \{p\}$ and $sel_1(q \mid s, I) = \{q\}$, and $sel_2(p \mid q, I) = \{q\}$ and $sel_2(q \mid s, I) = \{q\}$.

It is interesting to note that this head selection function $sel$ could be viewed as an analog of the Hilbert’s epsilon operator $\epsilon$ for the ASP scenario. Let $D = \{t_1, \cdots, t_m\}$ be the domain of a variable $x$. As mentioned in Introduction, $ex \ F$ returns some term $t$ in the domain $D$ such that $F(t)$ is true, otherwise it returns some default or arbitrary term; if there is more than one term making $F$ true, then any one of these terms can be chosen, nondeterministically. We can adapt $\epsilon$ to ASP as follows.

**Hilbert style epsilon operator for ASP:** Given an interpretation $I$, $ex \ F$ returns some term $t$ in the domain $D$ such that $F(t)$ is *satisfied by* $I$, otherwise it returns some default or arbitrary term *making* $F$ *false*; if there is more than one term making $F$ *satisfied by* $I$, then any one of these terms can be chosen, nondeterministically.

Therefore, given an interpretation $I$ the Hilbert quantification $F(ex \ F)$ nondeterministically selects one $F(t)$ satisfied by $I$ among $F(t_1), \cdots, F(t_m)$, and thus amounts to $sel(F(t_1) \mid \cdots \mid F(t_m), I)$.

As a result, the phrase “nondeterministically inferring a rule head formula $H_i$” amounts to “applying different head selection functions.” Therefore, given an interpretation, by applying a head selection function we can transform a disjunctive program into a normal program as follows.
**Definition 4 (disjunctive program reduct)** Let $\Pi$ be a disjunctive program, $I$ an interpretation and $\text{sel}$ a head selection function. The reduct of $\Pi$ w.r.t. $I$ and $\text{sel}$ is

$$\Pi^I_{\text{sel}} = \{ \text{sel}(\text{head}(r), I) \leftarrow \text{body}(r) \mid r \in \text{ground}(\Pi) \text{ such that } I \text{ satisfies } \text{body}(r) \}$$ (5)

Note that a reduct $\Pi^I_{\text{sel}}$ is a normal program and therefore we can apply any existing answer set semantics for normal programs to compute answer sets of $\Pi^I_{\text{sel}}$. Intuitively $I$ is a candidate answer set of $\Pi$ if $I$ is an answer set of $\Pi^I_{\text{sel}}$, and $I$ is an answer set of $\Pi$ if $I$ is minimal among all candidate answer sets. Formally we have the following definition.

**Definition 5 (general semantics for disjunctive programs)** Let $I$ be a model of a disjunctive program $\Pi$, and $\mathcal{X}$ be an answer set semantics for normal programs. Then $I$ is an answer set of $\Pi$ w.r.t. the base semantics $\mathcal{X}$ if (1) for some head selection function $\text{sel}$, $I$ is an answer set of $\Pi^I_{\text{sel}}$ under $\mathcal{X}$, and (2) $\Pi$ has no model $J \subset I$ satisfying condition (1).

The above semantics is well defined by its two conditions, where condition (1) defines a candidate answer set which is nondeterministically generated by means of head selection functions, while condition (2) enforces knowledge minimization by default negation, i.e., every answer set has a minimal number of atoms possible. In order to stress the intuition that candidate answer sets are nondeterministically determined by means of head selection functions by applying rules $H_1 \mid \cdots \mid H_k \leftarrow B$ such that when the body $B$ is satisfied we infer one $H_i$ from the head, we refer to the above semantics as a determining inference semantics, abbreviated as DI-semantics, for disjunctive programs.

In the following two sections, we illustrate some distinct properties of the DI-semantics by choosing the base semantics $\mathcal{X}$ to be the GL$_{nlp}$-semantics for simple normal programs (Gelfond and Lifschitz, 1988) and the well-justified answer set semantics for general normal programs (Shen et al., 2014), respectively.

### 4 Determining Inference Semantics for Simple Disjunctive Programs

In Definition 5, by replacing the base semantics $\mathcal{X}$ with the GL$_{nlp}$-semantics, we obtain a new answer set semantics for simple disjunctive programs.

**Definition 6 (DI-semantics for simple disjunctive programs)** Let $I$ be a model of a simple disjunctive program $\Pi$. Then $I$ is a DI-answer set of $\Pi$ if (1) for some head selection function $\text{sel}$, $I$ is an answer set of $\Pi^I_{\text{sel}}$ under GL$_{nlp}$-semantics, and (2) $\Pi$ has no model $J \subset I$ satisfying condition (1).

We use real world examples to illustrate the DL-semantics.

**Example 3** Consider the simple disjunctive program $\Pi$ in Example 1, where we showed

$$I = \{ \text{student}(\text{haoXue}), \text{majorInEnglish}(\text{haoXue}), \text{thesisInChinese}(\text{haoXue}), \text{thesisInEnglish}(\text{haoXue}) \}$$

is a desired answer set of $\Pi$, but it is not an answer set under the GL-semantics. We illustrate here that $I$ is a DI-answer set. The grounding $\text{ground}(\Pi)$ of $\Pi$ is as follows:

- $\text{thesisInChinese}(\text{haoXue}) \mid \text{thesisInEnglish}(\text{haoXue}) \leftarrow \text{student}(\text{haoXue})$
  - $r_1$
- $\text{student}(\text{haoXue})$  
  - $r_2$
- $\text{majorInEnglish}(\text{haoXue})$  
  - $r_3$
- $\text{thesisInChinese}(\text{haoXue})$  
  - $r_4$
- $f \leftarrow \text{majorInEnglish}(\text{haoXue}) \land \neg \text{thesisInEnglish}(\text{haoXue}) \land \neg f$  
  - $r_5$
Consider a head selection function \( \text{sel} \), where 
\[
\text{sel}(\text{head}(r_1), I) = \text{thesisInEnglish}(\text{haoXue}).
\]
By applying this selection we obtain the following reduct:

\[
\Pi_{\text{sel}}^I :
\begin{align*}
\text{thesisInEnglish}(\text{haoXue}) & \leftarrow \text{student}(\text{haoXue}) & r_1 \\
\text{student}(\text{haoXue}) & r_2 \\
\text{majorInEnglish}(\text{haoXue}) & r_3 \\
\text{thesisInChinese}(\text{haoXue}) & r_4
\end{align*}
\]

It is trivial to check that \( I \) is an answer set of \( \Pi_{\text{sel}}^I \) under the GL\_nlp-semantics, so condition (1) of Definition 6 is satisfied. As \( I \) is a minimal model of \( \Pi \), condition (2) is also satisfied. Thus \( I \) is a DI-answer set of \( \Pi \), which agrees with our expectation.

Next, we consider a generalization of the well-known Strategic Companies problem (Cadoli et al., 1997; Leone et al., 2006), which is popular for ASP benchmark competitions.

Example 4 (Generalized Strategic Companies problem (GSC)) Suppose a holding has companies \( C = \{c_1, \ldots, c_m\} \), and it produces goods \( G = \{g_1, \ldots, g_n\} \), where each company \( c_i \in C \) produces some goods \( G_i \subseteq G \). The holding wants to sell some of its companies subject to the following conditions: all products should be still in the portfolio and companies \( c_i \) for which a strategy rationale, expressed by justifications \( \sigma_{i1}, \ldots, \sigma_{ik_i} \) that are conjunctions \( \sigma_{ij} = \ell_{i1} \wedge \cdots \wedge \ell_{il_i} \) of literals on \( C \), holds true are not sold.

A set of companies \( C' \subseteq C \) constitutes a strategic set if (1) the companies in \( C' \) produce all the goods in \( G \), (2) in addition, only the companies \( c_i \) such that some justification \( \sigma_{ij} \) holds true relative to \( C' \) are in \( C' \) as well, and (3) \( C' \) is subset-minimal w.r.t. conditions (1) and (2). The classic Strategic Companies problem results if each justification \( \sigma_{ij} \) contains only positive literals, intuitively expressing that the companies occurring in it have the power to jointly control \( c_i \). To illustrate the generalized problem, consider companies \( C = \{c_1, c_2, c_3\} \) and goods \( G = \{g_1, g_2\} \), where \( g_1 \) can be produced either by \( c_1 \) or \( c_2 \), and \( g_2 \) produced by \( c_1 \) or \( c_3 \). Suppose that the strategy conditions are \( \sigma_1 = c_2 \wedge c_3 \), \( \sigma_2 = c_3 \), and \( \sigma_3 = c_1 \wedge \neg c_2 \). We may readily express this as a simple disjunctive program as follows:

\[
\Pi :
\begin{align*}
g_1 & , \ g_2 \\
c_1 & \leftarrow c_2 \leftarrow g_1 & r_1 \\
c_1 & \leftarrow c_3 \leftarrow g_2 & r_2 \\
c_1 & \leftarrow c_2 \wedge c_3 & r_3 \\
c_2 & \leftarrow c_3 & r_4 \\
c_3 & \leftarrow c_1 \wedge \neg c_2 & r_5
\end{align*}
\]

It is easy to check that \( \Pi \) has a DI-answer set \( I = \{g_1, g_2, c_1, c_2\} \), which corresponds to the single strategic set for this scenario. Again, this program has no GL-answer set.

Next we use the DI-semantics to formalize the well-known Barber paradox.

Example 5 (Russell’s Barber paradox) The barber is one who shaves all those, and those only, who do not shave themselves. The question is: does the barber shave himself? Obviously, answering this question results in a contradiction.

For simplicity of illustration, assume that there are only two men in the town, Bertrand and Dilbert, who need to be shaved, and Bertrand is the only barber. Then, this Barber paradox could be expressed by the following simple normal program.
\[ \Pi : \begin{align*}
&\text{man}(\text{Bertrand}) & r_1 \\
&\text{man}(\text{Dilbert}) & r_2 \\
&\text{barber}(\text{Bertrand}) & r_3 \\
&\text{shaves}(X,Y) \leftarrow \text{barber}(X) \land \text{man}(Y) \land \neg \text{shaves}(Y,Y) & r_4 
\end{align*} \]

Note that as the predicate \text{shaves}(\cdot) appears in the head of only one rule (i.e., \(r_4\)), in ASP this if-then rule amounts to an \textit{if and only if} (iff for short) rule, i.e., it represents an iff statement: “The barber \(X\) shaves a man \(Y\) iff \(Y\) does not shave himself.”

The grounding \(\text{ground}(\Pi)\) of \(\Pi\) is as follows:

\[ \begin{align*}
&\text{man}(\text{Bertrand}) & r_1 \\
&\text{man}(\text{Dilbert}) & r_2 \\
&\text{barber}(\text{Bertrand}) & r_3 \\
&\text{shaves}(\text{Bertrand},\text{Bertrand}) \leftarrow \text{barber}(\text{Bertrand}) \land \text{man}(\text{Bertrand}) \\
& & \quad \land \neg \text{shaves}(\text{Bertrand},\text{Bertrand}) & r_4 \\
&\text{shaves}(\text{Bertrand},\text{Dilbert}) \leftarrow \text{barber}(\text{Bertrand}) \land \text{man}(\text{Dilbert}) \\
& & \quad \land \neg \text{shaves}(\text{Dilbert},\text{Dilbert}) & r_5 \\
&\text{shaves}(\text{Dilbert},\text{Bertrand}) \leftarrow \text{barber}(\text{Dilbert}) \land \text{man}(\text{Bertrand}) \\
& & \quad \land \neg \text{shaves}(\text{Bertrand},\text{Bertrand}) & r_6 \\
&\text{shaves}(\text{Dilbert},\text{Dilbert}) \leftarrow \text{barber}(\text{Dilbert}) \land \text{man}(\text{Dilbert}) \\
& & \quad \land \neg \text{shaves}(\text{Dilbert},\text{Dilbert}) & r_7
\end{align*} \]

Due to \(r_4\) in \(\text{ground}(\Pi)\), this program has neither GL- nor DI-answer set.

Nevertheless, in order to unlock the Barber paradox to comply with our real world life, we may relax it a bit as: “The barber is one (1) who shaves all those who do not shave themselves and (2) who must shave someone (in the town).” The condition (2) seems to be a necessary requirement for a real life barber. Given that only \textit{Bertrand} and \textit{Dilbert} need to be shaved in the town, the condition (2) can be expressed using the following rule:

\[ \text{shaves}(X,\text{Bertrand}) \lor \text{shaves}(X,\text{Dilbert}) \leftarrow \text{barber}(X) \]

Intuitively, if \(X\) is a barber, then by applying \(r_8\) we nondeterministically infer either \(\text{shaves}(X,\text{Bertrand})\) meaning that \textit{Bertrand} is someone shaved by \(X\), or \(\text{shaves}(X,\text{Dilbert})\) meaning that \textit{Dilbert} is someone shaved by \(X\). Therefore, this relaxed version of the Barber paradox can be expressed by a simple disjunctive program \(\Pi' = \Pi \cup \{r_8\}\).

Due to the unique rule head atom requirement, \(\Pi'\) has no GL-answer set. We next show that it has a DI-answer set.

By applying \(r_2, r_3, r_5\) and \(r_8\), we can nondeterministically generate a potential answer set covering both \(\text{shaves}(\text{Bertrand},\text{Dilbert})\) and \(\text{shaves}(\text{Bertrand},\text{Bertrand})\) (i.e., we infer \(\text{shaves}(\text{Bertrand},\text{Bertrand})\) from \(r_8\) and infer \(\text{shaves}(\text{Bertrand},\text{Dilbert})\) from \(r_2, r_3, r_5\)). Consequently, we obtain the following DI-answer set for \(\Pi'\):

\[ I = \{\text{man}(\text{Bertrand}), \text{man}(\text{Dilbert}), \text{barber}(\text{Bertrand}), \text{shaves}(\text{Bertrand},\text{Dilbert}), \text{shaves}(\text{Bertrand},\text{Bertrand})\} \]

showing that barber Bertrand indeed shaves himself.

\[ \blacksquare \]

\textbf{Remark 2} In Example 5, we may use the Hilbert epsilon operator \(\epsilon\) to express rule \(r_8\) compactly as
\( shaves(X, \epsilon Y \ shaves(X,Y)) \leftarrow barber(X) \)

Recall that given a domain \( D = \{t_1, \cdots, t_m\} \) for a variable \( Y \), \( F(\epsilon Y \ F) \) nondeterministically selects one \( t \) from \( D \) for \( Y \) and thus amounts to \( F(t_1) \mid \cdots \mid F(t_m) \). In this sense, \( F(\epsilon Y \ F) \) could be viewed as a compact representation of \( F(t_1) \mid \cdots \mid F(t_m) \). This compact representation is significantly useful when \( m \) is a large number, even \( m = \infty \).

By condition (2) of Definition 6, a simple disjunctive program \( \Pi \) will never have two DI-answer sets \( I \) and \( J \) with \( J \subset I \). However, this does not mean that DI-answer sets of \( \Pi \) are minimal models of \( \Pi \), as shown in the following example.

**Example 6** Consider another simple disjunctive program:

\[
\Pi : \\
\begin{align*}
& a \\
& a \mid b \\
& c \leftarrow b \\
& d \leftarrow a \land \neg c \\
& r_1 \\
& r_2 \\
& r_3 \\
& r_4
\end{align*}
\]

Intuitively, by applying rules \( r_1 - r_3 \), we either infer \( a \) (i.e., infer \( a \) from \( r_1 \) and \( a \) from \( r_2 \)) or \( \{a, b, c\} \) (i.e., infer \( a \) from \( r_1 \), \( b \) from \( r_2 \) and \( c \) from \( r_3 \)). As a result, by applying rules \( r_1 - r_4 \) we infer two candidate answer sets: \( I_1 = \{a, d\} \) and \( I_2 = \{a, b, c\} \). The two candidates are minimal, thus we expect them to be answer sets of \( \Pi \).

It is easy to check that \( I_1 \) is an answer set under the GL-semantics, but \( I_2 \) is not. Next we show that both \( I_1 \) and \( I_2 \) are DI-answer sets of \( \Pi \).

For \( I_1 = \{a, d\} \), consider a head selection function \( sel \), where \( sel(a \mid b, I_1) = a \). By applying this selection we obtain the following reduct:

\[
\Pi_{sel}^{I_1} : \\
\begin{align*}
& a \\
& a \\
& d \leftarrow a \land \neg c \\
& r_1 \\
& r_2 \\
& r_3
\end{align*}
\]

\( I_1 \) is an answer set of \( \Pi_{sel}^{I_1} \) under the GL\_nlp-semantics, so condition (1) of Definition 6 is satisfied. As \( I_1 \) is a minimal model of \( \Pi \), condition (2) is also satisfied. Thus \( I_1 \) is a DI-answer set of \( \Pi \).

For \( I_2 = \{a, b, c\} \), consider a head selection function \( sel \), where \( sel(a \mid b, I_2) = b \). By applying this selection we obtain the following reduct:

\[
\Pi_{sel}^{I_2} : \\
\begin{align*}
& a \\
& b \\
& c \leftarrow b \\
& r_1 \\
& r_2 \\
& r_3
\end{align*}
\]

\( I_2 \) is an answer set of \( \Pi_{sel}^{I_2} \) under the GL\_nlp-semantics, so \( I_2 \) satisfies condition (1) of Definition 6. \( \Pi \) has only one model \( J = \{a, c\} \) with \( J \subset I_2 \). It is easy to check that for no head selection function \( sel \), \( J \) satisfies condition (1) of Definition 6. Therefore, \( I_2 \) also satisfies condition (2) and thus is a DI-answer set of \( \Pi \). Note that \( I_2 \) is not a minimal model of \( \Pi \).

As shown in the above examples, the DI-semantics does not have the unique rule head atom requirement and thus a DI-answer set is not necessarily an answer set under the GL-semantics.

For simple normal programs and positive simple disjunctive programs, the DI-semantics agrees with the GL-semantics.
Theorem 1 Let \( \Pi \) be a simple normal program or a positive simple disjunctive program. Then an interpretation \( I \) is a DI-answer set of \( \Pi \) if and only if \( I \) is an answer set under the GL-semantics.

Example 7 (Example 4 continued) In the case where a GSC instance is a classic Strategic Companies instance, i.e., each justification \( \sigma^3_i \) is a conjunction of positive literals, the rules for expressing the strategic sets similar as in Example 4 would be positive. For example, this is the case if the justification \( \sigma^3_1 \) would be \( c_1 \land c_2 \). Then, by replacing rule \( r_5 \) with \( c_3 \leftarrow c_1 \land c_2 \) we obtain a positive simple disjunctive program that amounts to the usual (propositional) Strategic Companies encoding. The DI-answer sets of that program are by Theorem 1 its GL-answer sets, which coincide for a positive program with its minimal models; for the modified justification \( \sigma^3_1 \) as above, we thus obtain the single DI-answer set \( I = \{g_1, g_2, c_1\} \), which is the single GL-answer set and the single minimal model of the program.

It is particularly interesting to observe that the GL-semantics can also be characterized using the disjunctive program reduct \( \Pi^I_{sel} \) of Definition 4 simply by requiring that for every (instead of some) head selection function \( sel \), \( I \) is an answer set of \( \Pi^I_{sel} \) under the GL-nlp-semantics. This reveals the essential difference of the determining inference semantics from the GL-semantics.

Theorem 2 Let \( I \) be a model of a simple disjunctive program \( \Pi \). Then \( I \) is an answer set of \( \Pi \) under the GL-semantics if and only if for every head selection function \( sel \), \( I \) is an answer set of \( \Pi^I_{sel} \) under the GL-nlp-semantics.

Let us consider some examples.

Example 8 (Example 3 continued) If we apply in Example 3 another head selection function \( sel' \), where \( sel'(head(r_1), I) = thesisInChinese(haoXue) \), we obtain the following reduct:

\[
\Pi^I_{sel'}: \quad thesisInChinese(haoXue) \leftarrow student(haoXue)
\quad student(haoXue)
\quad majorInEnglish(haoXue)
\quad thesisInChinese(haoXue)
\]

\( I \) is not an answer set of \( \Pi^I_{sel'} \) under the GL-nlp-semantics; thus by Theorem 2 \( I \) is not an answer set of \( \Pi \) under the GL-semantics.

Example 9 (Example 6 continued) In Example 6, for \( I_1 = \{a, d\} \), as there is only one selection function \( sel(a \mid b, I_1) = a \), by Theorem 2 \( I_1 \) is an answer set of \( \Pi \) under the GL-semantics. However, for \( I_2 = \{a, b, c\} \), if we apply another selection function \( sel' \) with \( sel'(a \mid b, I_2) = a \), we obtain the following reduct:

\[
\Pi^I_{sel'}: \quad a
\quad a
\quad c \leftarrow b
\]

\( I_2 \) is not an answer set of \( \Pi^I_{sel'} \) under the GL-nlp-semantics, so by Theorem 2 it is not an answer set of \( \Pi \) under the GL-semantics.

As shown in (Gelfond and Lifschitz, 1991), every answer set of a simple disjunctive program \( \Pi \) under the GL-semantics is a minimal model of \( \Pi \). Then the following corollary is immediate from Theorem 2.
Corollary 1 Let \( \Pi \) be a simple disjunctive program. If an interpretation \( I \) is an answer set of \( \Pi \) under the GL-semantics, then \( I \) is also a DI-answer set.

Finally we point out that a disjunctive rule head of the form \( a_1 \mid \cdots \mid a_m \) is essentially different from a choice construct (Simons et al., 2002; Ferraris and Lifschitz, 2005; Calimeri et al., 2012) of the form \( \{a_1, \cdots, a_m\} \), where \( 1 \leq u \leq m \). Given an interpretation \( I \) with \( S = I \cap \{a_1, \cdots, a_m\} \), then

- applying \( a_1 \mid \cdots \mid a_m \) we infer one atom \( a_i \in S \) if \( S \) is not empty, or \( \bot \) otherwise;
- applying \( \{a_1, \cdots, a_m\} \) we infer a set \( S \) of atoms if \( S \) is not empty and contains no more than \( u \) atoms, or \( \bot \) otherwise.

If \( u = 1 \), then applying \( \{a_1, \cdots, a_m\} \) enforces every answer set \( I \) to contain exactly one \( a_i \) in \( \{a_1, \cdots, a_m\} \), i.e., \( S = \{a_i\} \). In contrast, although applying \( a_1 \mid \cdots \mid a_m \) we infer only one atom \( a_i \), a DI-answer set \( I \) may well contain other atoms \( a_j \neq a_i \) in \( \{a_1, \cdots, a_m\} \), which are inferred by applying other rule heads.

If \( u > 1 \), then applying \( \{a_1, \cdots, a_m\} \) may lead to answer sets \( I \) and \( J \) with \( I \subset J \) and \( I \cap \{a_1, \cdots, a_m\} \subset J \cap \{a_1, \cdots, a_m\} \). In contrast, applying \( a_1 \mid \cdots \mid a_m \) will never lead to DI-answer sets \( I \) and \( J \) with \( I \subset J \).

For instance, in Example 4, (1) if we use choice constructs \( \{c_1, c_2\} \) and \( \{c_1, c_3\} \) to replace the heads of rules \( r_1 \) and \( r_2 \) respectively, we would obtain no answer set/strategic set from \( \Pi \); (2) if we use \( \{c_1, c_2\} \) and \( \{c_1, c_3\} \) to replace the rule heads, we would obtain two answer sets \( I = \{g_1, g_2, c_1, c_2\} \) and \( J = \{g_1, g_2, c_1, c_2, c_3\} \), which however violates the minimality condition of a strategic set.

5 Determining Inference Semantics for General Disjunctive Programs

General normal programs consist of rules of the form \( H \leftarrow B \), where \( H \) and \( B \) are first-order formulas. As shown in (Shen et al., 2014), existing answer set semantics for general normal programs such as those defined in (Pearce, 2006; Truszczynski, 2010; Bartholomew et al., 2011; Faber et al., 2011; Ferraris et al., 2011) suffer from the problem of circular justifications. To solve this issue, Shen et al. (2014) defined the well-justified answer set semantics whose answer sets are well justified by having a level mapping and thus are free of circular justifications. It is analogous to the GL\(_{nlp}\)-semantics for simple normal programs whose answer sets are well justified by having a level mapping (Fages, 1994). Therefore, in this section we present a DI-semantics for general disjunctive programs by replacing the base semantics \( \mathcal{X} \) in Definition 5 with the well-justified semantics.

The well-justified semantics is based on the one-step provability operator \( T_\Pi(O, N) \), which is an extension of the van Emde-Kowalski one-step provability operator (van Emde and Kowalski, 1976) from positive simple normal to general normal programs.

Definition 7 (Shen et al. (2014)) Let \( \Pi \) be a general normal program, and let \( O \) and \( N \) be two first-order theories. Define

\[
T_\Pi(O, N) = \{ \text{head}(r) \mid r \in \text{ground}(\Pi) \text{ and } O \cup N \models \text{body}(r) \}
\]

Informally, \( T_\Pi(O, N) \) collects all heads of rules in \( \text{ground}(\Pi) \) whose bodies are entailed by \( O \cup N \). When the parameter \( N \) is fixed, the entailment relation \( \models \) is monotone in \( O \), so \( T_\Pi(O, N) \) is monotone w.r.t. \( O \), i.e., for any first-order theories \( O_1 \subseteq O_2 \), we have \( T_\Pi(O_1, N) \subseteq T_\Pi(O_2, N) \). As moreover \( T_\Pi(O, N) \) is finitary,\(^5\) it is immediate that the inference sequence \( (T_\Pi(\emptyset, N))^i \)\(_{i=0}^\infty \), where \( T_\Pi(\emptyset, N) = \emptyset \) and for \( i \geq 0 \)

\(^5\)I.e., whenever \( \alpha \in T_\Pi(O, N) \), then there is some finite \( O' \subseteq O \) such that \( \alpha \in T_\Pi(O', N) \).
\( T^{i+1}_\Pi(\emptyset, N) = T^i_\Pi(T^i_\Pi(\emptyset, N), N) \), will converge to a least fixpoint, denoted \( \text{lfp}(T^i_\Pi(\emptyset, N)) \).

The well-justified FLP answer set semantics, abbreviated as well-justified semantics or WJ-semantics, is defined in terms of the least fixpoint \( \text{lfp}(T^i_\Pi(\emptyset, \neg I^-)) \) w.r.t. an interpretation \( I \).

**Definition 8 (Shen et al. (2014))** Let \( I \) be a model of a general normal program \( \Pi \). Then \( I \) is a WJ-answer set of \( \Pi \) if \( \text{lfp}(T^i_\Pi(\emptyset, \neg I^-)) \cup \neg I^- \models A \) for every \( A \in I \).

Then, by replacing the base semantics \( \mathcal{X} \) in Definition 5 with the WJ-semantics we obtain a DI-answer set semantics for general disjunctive programs.

**Definition 9 (DI-semantics for general disjunctive programs)** Let \( I \) be a model of a general disjunctive program \( \Pi \). Then \( I \) is a DI-answer set of \( \Pi \) if (1) for some head selection function \( \text{sel} \), \( I \) is a WJ-answer set of \( \Pi_{\text{sel}}^I \), and (2) \( \Pi \) has no model \( J \subset I \) satisfying condition (1).

Intuitively, a DI-answer set of a general disjunctive program is a model that is minimal among all of the models that can be nondeterministically (by means of a head selection function) inferred by iteratively applying rules via a bottom up fixpoint sequence.

It was proved in (Shen et al., 2014) that WJ-answer sets of a general normal program \( \Pi \) are minimal models of \( \Pi \). Then the following result is immediate from Definition 9.

**Corollary 2** For a general normal program \( \Pi \), an interpretation \( I \) is a DI-answer set of \( \Pi \) if and only if \( I \) is a WJ-answer set of \( \Pi \).

As shown in (Shen et al., 2014), for simple normal programs the WJ-semantics coincides with the \( \text{GL}_{\text{nlp}} \)-semantics. Then, for a simple disjunctive program \( \Pi, \Pi_{\text{sel}}^I \) is a simple normal program and thus \( I \) is a WJ-answer set of \( \Pi_{\text{sel}}^I \) iff \( I \) is an answer set of \( \Pi_{\text{sel}}^I \) under the \( \text{GL}_{\text{nlp}} \)-semantics. Therefore, the following result is immediate.

**Corollary 3** For a simple disjunctive program \( \Pi \), an interpretation \( I \) is a DI-answer set under Definition 9 if and only if \( I \) is a DI-answer set under Definition 6.

We use simple examples to illustrate the DI-semantics for general disjunctive programs.

**Example 10 (Example 2 continued)** The grounding \( \text{ground}(\Pi) \) of \( \Pi \) is as follows:

\[
\begin{align*}
mammal(grus\_grus) & \supset \text{vertebrate(grus\_grus)} \mid \text{vertebrate(grus\_grus)} \\
& \supset \text{mammal(grus\_grus)} \leftarrow \text{animal(grus\_grus)} \quad (r_1) \\
\text{animal(grus\_grus)} \quad & (r_2) \\
mammal(grus\_grus) \quad & (r_3) \\
\bot & \leftarrow \neg \text{vertebrate(grus\_grus)} \quad (r_4)
\end{align*}
\]

\( I = \{ \text{animal(grus\_grus)}, \text{mammal(grus\_grus)}, \text{vertebrate(grus\_grus)} \} \) is the only model of \( \Pi \). Consider a head selection function \( \text{sel} \), where

\[
\text{sel}(\text{head}(r_1), I) = \text{mammal(grus\_grus)} \supset \text{vertebrate(grus\_grus)}
\]

By applying this selection we obtain the following reduct:

\[
\Pi_{\text{sel}}^I : \text{mammal(grus\_grus)} \supset \text{vertebrate(grus\_grus)} \leftarrow \text{animal(grus\_grus)} \quad (r_1) \\
\text{animal(grus\_grus)} \quad & (r_2) \\
\text{mammal(grus\_grus)} \quad & (r_3)
\]
The least fixpoint
\[ \text{lfp}(T_{\Pi \set}^I (\emptyset, \neg I^-)) = \{ \text{animal(grus.grus), mammal(grus.grus),} \]
\[ \text{mammal(grus.grus) \supset vertebrate(grus.grus)} \}
\]
is obtained from the following inference sequence:
\[ T_{\Pi \set}^0 (\emptyset, \neg I-) = \emptyset, \]
\[ T_{\Pi \set}^1 (\emptyset, \neg I-) = \{ \text{animal(grus.grus), mammal(grus.grus)} \} \text{ (by } r_2, r_3). \]
\[ T_{\Pi \set}^2 (\emptyset, \neg I-) = \{ \text{animal(grus.grus), mammal(grus.grus),} \]
\[ \text{mammal(grus.grus) \supset vertebrate(grus.grus)} \} \text{ (by } r_1, r_2, r_3). \]

As \( \text{lfp}(T_{\Pi \set}^I (\emptyset, \neg I^-)) \cup \neg I^- \) entails every atom in \( I \), \( I \) is a WJ-answer set of \( \Pi \set \), so condition (1) of Definition 9 is satisfied. As \( I \) is a minimal model of \( \Pi \), condition (2) of Definition 9 is also satisfied. Hence \( I \) is a DI-answer set of \( \Pi \), as we expected. \( \square \)

The following example shows that a DI-answer set is not necessarily a minimal model.

**Example 11** Consider the following general disjunctive program:

\[ \Pi : \begin{align*}
\text{a} \land q & \mid \neg a & r_1 \\
\text{p} & \leftarrow q & r_2 \\
\text{q} & \leftarrow \neg q \lor p & r_3
\end{align*} \]

It has two models: \( I_1 = \{ p, q \} \) and \( I_2 = \{ a, p, q \} \).

For \( I_1 \) there is only one head selection function \( \text{sel} \), where \( \text{sel}(a \land q \mid \neg a, I_1) = \neg a \). By applying this selection we obtain the following reduct:

\[ \Pi_{\text{sel}}^{I_1} : \begin{align*}
\neg a & \quad r_1 \\
p & \leftarrow q & r_2 \\
q & \leftarrow \neg q \lor p & r_3
\end{align*} \]

The least fixpoint \( \text{lfp}(T_{\Pi \set}^I (\emptyset, \neg I_1^-)) = \{ \neg a \} \) is obtained from the following inference sequence:
\[ T_{\Pi \set}^0 (\emptyset, \neg I_1^-) = \emptyset, \]
\[ T_{\Pi \set}^1 (\emptyset, \neg I_1^-) = \{ \neg a \} \text{ (by } r_1). \]

As \( \text{lfp}(T_{\Pi \set}^I (\emptyset, \neg I_1^-)) \cup \neg I_1^- \nleq p \), \( I_1 \) is not an answer set of \( \Pi \set \) under the WJ-semantics, so condition (1) of Definition 9 is not satisfied and thus \( I_1 = \{ p, q \} \) is not a DI-answer set of \( \Pi \).

Next consider the model \( I_2 = \{ a, p, q \} \). There is only one head selection function \( \text{sel} \), where \( \text{sel}(a \land q \mid \neg a, I_2) = a \land q \), applying which yields the following reduct:

\[ \Pi_{\text{sel}}^{I_2} : \begin{align*}
a \land q & \quad r_1 \\
p & \leftarrow q & r_2 \\
q & \leftarrow \neg q \lor p & r_3
\end{align*} \]

The least fixpoint \( \text{lfp}(T_{\Pi \set}^I (\emptyset, \neg I_2^-)) = \{ a \land q, p, q \} \) is obtained from the following inference sequence:
As $lfp(T_{\Pi_{sel}^I}(\emptyset, -I^-)) = \emptyset$, $T_{\Pi_{sel}^I}(\emptyset, -I^-) = \{a \land q\}$ (by $r_1$), $T_{\Pi_{sel}^I}(\emptyset, -I^-) = \{a \land q, p\}$ (by $r_1, r_2$), $T_{\Pi_{sel}^I}(\emptyset, -I^-) = \{a \land q, p, q\}$ (by $r_1, r_2, r_3$).

As $lfp(T_{\Pi_{sel}^I}(\emptyset, -I^-)) \cup \neg I^- \models a \land p \land q$, $I_2$ is an answer set of $\Pi_{sel}^I$ under the WJ-semantics, so condition (1) of Definition 9 is satisfied. As $I_1$ is the only model of $\Pi$ which is a proper subset of $I_2$ and $I_1$ does not satisfy condition (1), $I_2$ satisfies condition (2) of Definition 9. Therefore, $I_2$ is a DI-answer set of $\Pi$.

**Remark 3** In Example 11, it is not appropriate to admit $I_1 = \{p, q\}$ as an answer set of $\Pi$ because it has a circular justification via the self-supporting loop

$$p \leftarrow q \leftarrow \neg q \lor p \leftarrow p$$

where the arrow $\leftarrow$ stands for “is due to”; i.e., $p$ being true in $I_1$ is due to $q$ being true in $I_1$ (via rule $r_2$) that is due to $I_1$ satisfying $\neg q \lor p$ (via rule $r_3$), which in turn is due to $p$ being true in $I_1$.

The following example further illustrates that the DI-semantics clearly differentiates the two inference operators $\leftarrow$ and $|$ from the two logical connectives $\supset$ and $\lor$.

**Example 12** Consider the following disjunctive program:

$$\Pi_1 : p \mid q$$

$$p \leftarrow q$$

$$q \leftarrow p$$

Intuitively, by applying $r_1$ we nondeterministically infer $p$ or $q$; whichever we choose, by applying the three rules we always infer $I = \{p, q\}$. Note that $I$ is a minimal model of $\Pi_1$. We next show that $I$ is a DI-answer set of $\Pi_1$. Consider a head selection function with $sel(p \mid q, I) = p$. We have the following reduct:

$$\Pi_{sel}^I : p$$

$$p \leftarrow q$$

$$q \leftarrow p$$

$I = \{p, q\}$ is an answer set of $\Pi_{sel}^I$ under the WJ-semantics, where we obtain that the least fixpoint is given by $lfp(T_{\Pi_{sel}^I}(\emptyset, -I^-)) = \{p, q\}$. Thus $I$ is a DI-answer set of $\Pi_1$.

Next, let us consider the following variant of $\Pi_1$, where the inference operator $\mid$ in $r_1$ is replaced by the logical disjunction connective $\lor$:

$$\Pi_2 : p \lor q$$

$$p \leftarrow q$$

$$q \leftarrow p$$

Intuitively, by applying $r_1$ we infer $p \lor q$; as neither of the bodies of $r_2$ and $r_3$ is entailed by $p \lor q$, applying the three rules only infers $p \lor q$. As neither $p$ nor $q$ is entailed by $p \lor q$, $I = \{p, q\}$ should not be an answer set of $\Pi_2$.

We next show that $I = \{p, q\}$ is not a DI-answer set of $\Pi_2$. For every rule $r$ in $\Pi_2$ there is only one head selection function $sel$ on $I$, where $sel(head(r), I) = head(r)$. We have the following reduct:
$$\Pi^I_{2sel}: p \lor q$$
$$p \leftarrow q$$
$$q \leftarrow p$$

$I = \{p, q\}$ is not an answer set of $\Pi^I_{2sel}$ under the WJ-semantics, where the least fixpoint is given by $\operatorname{lfp}(T_{\Pi^I_{2sel}}(\emptyset, \neg I^-)) = \{p \lor q\}$. Thus $I$ is not a DI-answer set of $\Pi_2$.

Finally, consider the following variant of $\Pi_2$, where the inference operator $\leftarrow$ in $r_2$ and $r_3$ is replaced by the logical implication connective $\supset$:

$$\Pi_3: p \lor q$$
$$q \supset p$$
$$p \supset q$$

Intuitively, as $\Pi_3$ is a first-order theory, applying its rules infers $\Pi_3$ itself. As both $p$ and $q$ are entailed by $\Pi_3$, $I = \{p, q\}$ is expected to be an answer set of $\Pi_3$.

We next show that $I = \{p, q\}$ is a DI-answer set of $\Pi_3$. There is only one head selection function $\text{sel}$ on $I$, thus leading to the following reduct:

$$\Pi^I_{3sel}: p \lor q$$
$$q \supset p$$
$$p \supset q$$

$I = \{p, q\}$ is an answer set of $\Pi^I_{3sel}$ under the WJ-semantics, where for the least fixpoint, we obtain that $\operatorname{lfp}(T_{\Pi^I_{3sel}}(\emptyset, \neg I^-)) = \Pi^I_{3sel}$. Thus $I$ is a DI-answer set of $\Pi_3$. ■

### 6 Computational Complexity

When all rules of a general normal program have an empty body, this program amounts to a first-order theory. As it is undecidable to determine whether an arbitrary first-order theory is satisfiable, it is undecidable to determine whether a general disjunctive program has a DI-answer set w.r.t. any base answer set semantics for general normal programs. Therefore, we address in this paper the computational complexity of propositional logic programs under Herbrand interpretations, and we focus on the DI-semantics with the well-justified semantics (Shen et al., 2014) as the base semantics and refer to it as DI-WJ answer set semantics; we briefly comment on other base semantics in Section 6.1.

As usual, we consider three canonical decision problems:

Table 1: Complexity of the DI-WJ answer set semantics for propositional logic programs (entries denote completeness)

<table>
<thead>
<tr>
<th>Program \ Problem</th>
<th>Answer set existence</th>
<th>Cautious reasoning</th>
<th>Brave reasoning</th>
</tr>
</thead>
<tbody>
<tr>
<td>general disjunctive</td>
<td>$\Sigma_2^p$</td>
<td>$\Pi_3^p$</td>
<td>$\Sigma_3^p$</td>
</tr>
<tr>
<td>general normal</td>
<td>$\Sigma_2^p$ *</td>
<td>$\Pi_2^p$ *</td>
<td>$\Sigma_2^p$ *</td>
</tr>
<tr>
<td>simple disjunctive</td>
<td>$\text{NP}$</td>
<td>$\Pi_2^p$</td>
<td>$\Sigma_2^p$</td>
</tr>
<tr>
<td>simple normal</td>
<td>$\text{NP}$ *</td>
<td>$\text{co-NP}$ *</td>
<td>$\text{NP}$ *</td>
</tr>
</tbody>
</table>

* Results that are immediate from Theorem 2 and the complexity results in (Shen et al., 2014).
(1) *answer set existence*, i.e., the problem of deciding whether a given logic program \( \Pi \) has some DI-answer set;

(2) *cautious reasoning*, i.e., the problem of deciding whether a ground literal \( L \) is true in all DI-answer sets of \( \Pi \); and

(3) *brave reasoning*, i.e., the problem of deciding whether a ground literal \( L \) is true in some DI-answer set of \( \Pi \).

The complexity results are compactly summarized in Table 1. Besides the general case, also some of the syntactic fragments that we have discussed are considered. As can be seen from the table, the occurrence of determined selection increases the complexity of the reasoning problems, except for answer set existence, by one level in the polynomial hierarchy. If we restrict the literal \( L \) to a ground atom, then the complexity of cautious reasoning for general disjunctive programs drops to \( \Pi_2^p \), but stays the same in all other cases.

The results are derived as follows. By Corollary 2, the well-justified semantics from (Shen *et al.*, 2014) coincides for general normal programs with the DI-WJ answer set semantics; the entries marked with an asterisk * in Table 1 are thus justified by the complexity results in that paper.

Next, the results for simple disjunctive programs are easily obtained from the fact that DI-WJ answer sets and GL-answer sets coincide for simple normal programs and simple positive programs by Theorem 1, and by the well-known complexity results for the decision problems that we consider shown in (Eiter and Gottlob, 1995) for these program classes.

**Theorem 3**

Given a propositional simple disjunctive program \( \Pi \) and a ground literal \( L \), deciding (i) whether \( \Pi \) has some DI-WJ answer set is NP-complete, (ii) whether \( L \) is true in every DI-WJ answer set of \( \Pi \) is \( \Pi_2^p \)-complete, and (iii) whether \( L \) is true in some DI-WJ answer set of \( \Pi \) is \( \Sigma_2^p \)-complete.

It remains to justify the entries for general disjunctive programs. To this end, we first note the following lemma.

**Lemma 1**

Given a propositional general disjunctive program \( \Pi \) and an interpretation \( I \) of \( \Pi \), deciding whether condition (1) of Definition 9 holds is in \( \Sigma_2^p \).

Intuitively, to verify that condition (1) of Definition 9 holds we need to guess a selection \( sel \) witnessing that \( I \) is a well-justified answer set of \( \Pi_{sel} \); the latter check is in co-NP; overall, this yields \( \Sigma_2^p \) membership.

For DI-WJ answer set existence, condition (2) in Definition 9 is irrelevant. We thus establish the following result.

**Theorem 4**

Given a propositional general disjunctive program \( \Pi \), deciding whether \( \Pi \) has some DI-WJ answer set is \( \Sigma_2^p \)-complete.

Lemma 1 allows us furthermore to derive the following bound on DI-WJ answer set checking, i.e., deciding whether a given model of a general disjunctive program is a well-justified answer set.

**Proposition 1**

Given a propositional general disjunctive program \( \Pi \) and an interpretation \( I \) of \( \Pi \), deciding whether \( I \) is a DI-WJ answer set of \( \Pi \) is in \( D_2^p \) (thus in \( \Pi_{\Sigma_2^p} \)).
to true and false, respectively. Intuitively, the $\Sigma_2^p$-component is here the same as in Lemma 1 and the $\Pi_2^p$ component is to check that no smaller $J \subset I$ satisfies condition (1), which is in $\text{co-}\Sigma_2^p = \Pi_2^p \phone$. (In fact, DI-WJ answer set checking can also be shown to be $D_2^p$-hard, by a construction derived from the proof of Theorem 5 below, and is thus $D_2^p$-complete.)

As a consequence, brave reasoning has a $\Sigma_2^p$ upper bound, and dually cautious reasoning a $\Pi_2^p$ upper bound. The matching lower bound can be shown by a polynomial reduction from the following problem MINQASAT: given a QBF $\forall Y.E(X, Y)$, where $X$ and $Y$ are lists (sets) of Boolean variables, and an atom $A$ in $X$, is there some minimal variable assignment $\sigma$ to $X$ (viewed as model $\{X_i \in X \mid \sigma(X_i) = 1\}$), such that $\forall Y.E(\sigma(X), Y)$ evaluates to true and $A$ is true in $\sigma$?

**Lemma 2** Deciding MINQASAT is $\Sigma_3^p$-complete.

**Theorem 5** Given a propositional logic program $\Pi$ and a ground literal $L$, deciding whether $L$ is true in some (resp. every) well-justified answer set of $\Pi$ is $\Sigma_3^p$-complete (resp. $\Pi_3^p$-complete).

We note that by restricting the literal $L$ in Theorem 5 to a ground atom, cautious reasoning is in $\Pi_2^p$; the reason is that an interpretation $I$ of the program $\Pi$ that fulfills condition (1) of Definition 9 but does not satisfy $L$ suffices to refute the query; Lemma 1 implies that deciding the existence of such a model $I$ is in $\Sigma_2^p$.

The higher computational complexity of brave and cautious reasoning in the general disjunctive case, compared to GL-answer sets or the well-known FLP-answer sets defined by Faber et al. (2011) (where the operator $|$ is viewed as the logical connective $\lor$ for which the problems are $\Sigma_2^p$-complete and $\Pi_2^p$-complete, respectively, offers higher problem solving capacity. However, even if some complex formulas are used in a general disjunctive program, the reasoning complexity may stay within $\Sigma_2^p$ resp. $\Pi_2^p$. For example, if in the encoding of strategic sets in Example 4 the justifications $\sigma_j^i$ were monotone DNFs $\sigma_j^i = D_1 \lor \cdots \lor D_k$, the problem of deciding relevance of a given company $c_i$ for strategic sets, i.e., whether $c_i$ occurs in some strategic set remains in $\Sigma_2^p$; this is because the rule $c_i \leftarrow \sigma_j^i$ can be replaced with rules $c_i \leftarrow D_1, \ldots, c_i \leftarrow D_k$, such that we obtain a positive simple disjunctive program. Note that the strategic sets which are given by the DI-answer sets (equivalently, the GL-answer sets resp. minimal models) of this program, are free from self-support and circular justification. If the justifications $\sigma_j^i$ would be arbitrary propositional formulas and we are interested in strategic sets with that property, we can consider the DI-WJ answer sets of the respective general disjunctive program, which encode such strategic sets. As can be seen, deciding relevance of a company $c_i$ for strategic sets is $\Sigma_2^p$-complete in this setting; this could not be expressed with a disjunctive program under the GL-answer set or the FLP-answer set semantics.

**Computing some DI-WJ answer set**

Furthermore, we note that computing some DI-WJ answer set of a simple normal respectively general normal program $\Pi$ can be seen to be complete for the (multi-valued) functional analogs of NP and $\Sigma_2^p$, respectively.\(^6\)

Notably, computing the DI-WJ answer sets of a general disjunctive program $\Pi$ can be reduced to computing the minimal models of a QBF of the form $\exists Z \forall Y.E(X, Y, Z)$ that is constructible in polynomial time. To this end, one can express condition (1) in Definition 9 as a QBF of this form in polynomial time, where the free variables $X$ describe $I$. The converse direction is also possible; the QBF reduction in the proof of Theorem 5 can be extended for the “$\exists Z$” part such that the minimal models of $\Phi$ correspond one-to-one to

\(^6\)The completeness is in the sense of strong computability, i.e. that all (and only) the solutions to the input instance are computed by a nondeterministic transducer without resp. with an NP oracle access.
the DI-WJ answer sets of the constructed program $\Pi_n$. Thus, computing the DI-WJ answer sets of a general disjunctive program $\Pi$ and the minimal models of a QBF $\exists Z \forall Y. E(X, Y, Z)$ are polynomial-time equivalent problems; this analogously holds for simple disjunctive programs and QBFs $\exists Z. E(X, Y, Z)$.

Computing some minimal model of a QBF $\exists Z. E(X, Z)$ (resp., $\exists Z \forall Y. E(X, Y, Z)$) is possible using an oracle for NP (resp. $\Sigma^p_2$) in polynomial time. In fact, a bounded number of calls to a witness oracle for NP (resp. $\Sigma^p_2$) is sufficient, i.e., an oracle that not only answers “yes” or “no” to a query, but provides in the “yes” case also a polynomial-size witness to the query; e.g. in case of a SAT oracle, this is some satisfying assignment, and in case of an $\exists X \forall Y. E(X, Y)$ oracle some assignment $\sigma$ to $X$ such that $E(\sigma(X), Y)$ is a tautology.

In fact, computing some minimal model of a QBF $\exists Z. E(X, Z)$ is complete for the class $\text{FP}^{\text{NP}}[\log, \text{wit}]$, which contains the class of multi-valued functions $f$ for which some possible function value $y \in f(x)$ is computable in polynomial time with logarithmically many calls to a witness oracle in NP (Buss et al., 1993). This result, which in essence was shown by Chen and Toda (1995) – who considered a class $\text{FNP}^{\text{OptP}}[\log n]$ that coincides with $\text{FP}^{\text{NP}}[\log, \text{wit}]$ on total multi-valued functions–, carries by the polynomial-time equivalence over to computing some DI-WJ answer set of a simple disjunctive program $\Pi$.

Based on techniques in (Chen and Toda, 1995), one can show that computing some minimal model of a QBF $\exists X \forall Y. E(X, Y)$ is complete for the class $\text{FP}^{\Sigma^p_2}[\log, \text{wit}]$, which is analogous to $\text{FP}^{\Sigma^p_2}[\log, \text{wit}]$ but allows for a $\Sigma^p_2$ witness oracle. Consequently, computing some DI-WJ answer set of a general disjunctive program $\Pi$ is $\text{FP}^{\Sigma^p_2}[\log, \text{wit}]$-complete.

For more details on witness oracles, we refer to (Janota and Marques-Silva, 2016; Endriss and de Haan, 2015) and references therein (see also appendix).

6.1 DI-answer sets for other base semantics

If in Definition 5 we take the GL-semantics as the base semantics, then we obtain for simple normal and simple disjunctive programs the same complexity results as for DI-WJ answer sets in Table 1; this is an immediate consequence of the fact that for simple normal programs, the GL$_{wlp}$-answer sets, the GL-answer sets and the well-justified answer sets coincide as shown by Shen et al. (2014); hence the DI-answer sets coincide under the three base semantics. The same complexity results hold if we take the FLP answer set semantics as the base semantics, as the FLP-answer sets coincide with the GL-answer sets on simple normal programs. Note that deciding whether a propositional simple disjunctive program $\Pi$ has DI-answer sets is NP-complete, which is lower than deciding whether $\Pi$ has GL-answer sets; the latter is $\Sigma^p_2$-complete (Eiter and Gottlob, 1995).

For the class of general normal and general disjunctive programs (which was not considered by Gelfond and Lifschitz), the results in Table 1 also hold if we take the FLP-semantics as the base semantics. In fact, the results for general normal programs have been established by Shen et al. (2014); the membership parts for general disjunctive programs can be shown by simple guess-and-check algorithms, and the matching hardness results carry over from simple disjunctive programs (for answer set existence) and the proof of Theorem 5, as the reduction from QBFs given there works for FLP-semantics as the base semantics as well.

7 Related Work

The seminal work of ASP is by Gelfond and Lifschitz (1988, 1991), who considered simple normal and simple disjunctive programs, respectively and defined the seminal answer set semantics (i.e., the GL-semantics)
for them. In the late 1990s, answer set programming was identified as a new declarative problem solving paradigm by Lifschitz (1999); Marek and Truszczyński (1999); Niemela (1999). Since then, various extensions have been presented to expand the expressive power and applicability of ASP, including logic programs with aggregates (Pelov, 2004; Shen and You, 2007; Son et al., 2007; Faber et al., 2011; Asuncion et al., 2015), with external sources such as description logic programs (Eiter et al., 2008), with propositional or first-order formulas (Pearce, 2006; Pelov et al., 2007; Truszczyński, 2010; Bartholomew et al., 2011; Ferraris et al., 2011; Shen et al., 2014), and with epistemic negations (Gelfond, 2011; Truszczyński, 2011; Kahl, 2014; Shen and Eiter, 2016). All of these extensions take the GL-semantics as the standard answer set semantics for simple disjunctive programs.

We mention two representative related answer set semantics for disjunctive programs: the FLP-semantics by Faber et al. (2011) and the Equilibrium logic-based semantics in (Ferraris, 2005; Pearce, 2006; Ferraris et al., 2011). For illustration of their differences, we present the following example.

**Example 13** Consider the following disjunctive programs:

\[
\Pi_1: \begin{align*} & p \lor \neg p \\ & p \leftarrow \neg \neg p \\
\end{align*}
\]

\[
\Pi_2: \begin{align*} & p \lor \neg p \\ & p \leftarrow \neg \neg p \\
\end{align*}
\]

\(\Pi_1\) differs from \(\Pi_2\) in that by applying \(r_1\) we nondeterministically infer \(p\) or \(\neg p\), while by applying \(r_1'\) we infer \(p \lor \neg p\).

For \(\Pi_1\), \(r_1\) generates two potential answer sets \(I_1 = \{p\}\) and \(I_2 = \emptyset\), and together with the constraint \(r_2\) (i.e., \(\neg p\) cannot occur in any answer set) yields a unique candidate answer set \(I_1\). As \(I_1\) is a minimal model of \(\Pi_1\), it is a desired answer set of \(\Pi_1\).

In contrast, as \(p \lor \neg p\) is a tautology, \(\Pi_2\) amounts to the program \(\{\top, p \leftarrow \neg \neg p\}\) and thus should have no answer set.

It is easy to check that \(I_1 = \{p\}\) is a DI-answer set of \(\Pi_1\) and \(\Pi_2\) has no DI-answer set.

### 7.1 FLP-semantics

The FLP-semantics (Faber et al., 2004, 2011) was originally oriented towards giving an answer set semantics to simple disjunctive programs extended with aggregates; it was later deployed to other extensions of logic programs, including description logic programs (Eiter et al., 2005, 2008; Lukasiewicz, 2010), modular logic programs (Dao-Tran et al., 2009), and logic programs with first-order formulas (Bartholomew et al., 2011). It is straightforward to lift the FLP-semantics from simple disjunctive programs to general disjunctive programs as follows. Let \(\Pi\) be a general disjunctive program and \(I\) an interpretation; then \(I\) is an FLP-answer set of \(\Pi\) if \(I\) is a minimal model of the FLP-reduct

\[
f\Pi^I = \{r \in \text{ground}(\Pi) \mid I \text{ satisfies body}(r)\}
\]

of \(\Pi\) w.r.t. \(I\).

In contrast to the DI-semantics, the FLP-semantics has three major limitations:

1. For simple disjunctive programs, it coincides with the GL-semantics and thus may exclude some desired answer sets.
(2) It does not distinguish between the nondeterministic inference operator $|$ and the logical disjunction connective $\lor$. Specifically, it identifies $|$ with $\lor$. For example, the two disjunctive programs in Example 13 are the same under the FLP-semantics and have no FLP-answer set.

(3) Its answer sets may have circular justifications that are caused by self-supporting loops. As a result, an FLP-answer set may not be a DI-answer set and vice versa. For instance, in Example 11, $I_1 = \{p, q\}$ is an FLP-answer set, but it is not a DI-answer set because it has a circular justification; however, $I_2 = \{a, p, q\}$ is a DI-answer set, but it is not an FLP-answer set.

7.2 Equilibrium logic-based semantics

Another extensively studied answer set semantics is by Ferraris (2005), which defines answer sets for logic programs with propositional formulas and aggregates based on a new definition of equilibrium logic (Pearce, 1996). This semantics was further extended to first-order formulas in terms of a modified circumscription (Ferraris et al., 2011). Following a similar way, Pearce (2006) proposed to identify answer sets with equilibrium models in equilibrium logic. It turns out that the answer set semantics of Pearce (2006) coincides with that of Ferraris (2005) in the propositional case and with that of Ferraris et al. (2011) in the first-order case. Therefore, we refer to them as Equilibrium logic-based semantics.

For any ground formula $F$, the Ferraris-reduct of $F$ w.r.t. an interpretation $I$, denoted $F^I_L$, is defined recursively as follows (Ferraris, 2005):

- $\bot_I = \bot$
- $A^I_L = \begin{cases} A & \text{if } A \text{ is a ground atom and } I \text{ satisfies } A \\ \bot & \text{otherwise} \end{cases}$
- $(\neg F)^I_L = \begin{cases} \bot & \text{if } I \text{ satisfies } F \\ \top & \text{otherwise} \end{cases}$
- $(G \circledast H)^I_L = \begin{cases} G^I_L \circledast H^I_L & \text{if } I \text{ satisfies } G \circledast H \ (\circledast \in \{\land, \lor, \supset\}) \\ \bot & \text{otherwise} \end{cases}$

Let $\Pi$ be a propositional program and $\Pi'$ be $\Pi$ with $|$ and $\leftarrow$ replaced by $\lor$ and $\subset$, respectively. Then $I$ is an answer set of $\Pi$ under the Equilibrium logic-based semantics if $I$ is a minimal model of the Ferraris-reduct $\Pi'_L$ of $\Pi'$.

The Equilibrium logic-based semantics has the above three major limitations as the FLP-semantics, but it does not treat $p \lor \neg p$ as a tautology. As a result, the two disjunctive programs in Example 13 have the same answer set $I = \{p\}$ under the Equilibrium logic-based semantics.

Truszczynski (2010) also presented a reduct that slightly differs from the Ferraris-reduct in the way to handle $(G \supset H)^I_L$, i.e.,

- $(G \supset H)^I_L = \begin{cases} G \supset H^I_L & \text{if } I \text{ satisfies both } G \text{ and } H \\ \top & \text{if } I \text{ does not satisfy } G \\ \bot & \text{otherwise} \end{cases}$

This semantics also inherits the above three major limitations.

7.3 Split programs

For a simple disjunctive program $\Pi$, Hitzler and Seda (1999) proposed to split $\Pi$ into a collection of simple normal programs, called normal derivatives. Informally, a normal derivative $P(\Pi)$ of $\Pi$ is obtained from
ground(\(\Pi\)) by replacing every disjunctive rule \(A_1 \cdots A_k \leftarrow body(r)\) (where \(k \geq 2\)) arbitrarily with one or more rules of the form \(A_i \leftarrow body(r), 1 \leq i \leq k\). For example, take \(\Pi = \{p \leftarrow q \leftarrow \neg s\}\); then \(\Pi\) has three normal derivatives: \(P_1(\Pi) = \{p \leftarrow \neg s\}, P_2(\Pi) = \{q \leftarrow \neg s\}\), and \(P_3(\Pi) = \{p \leftarrow \neg s, q \leftarrow \neg s\}\).

Hitzler and Seda (1999) attempted to use the normal derivatives to characterize the GL-semantics of a simple disjunctive program \(\Pi\) and showed that if an interpretation \(I\) is an answer set of \(\Pi\) under the GL-semantics, then \(I\) is an answer set of some normal derivative of \(\Pi\) under the GL\(_{nlp}\)-semantics. Take the above logic program \(\Pi: I = \{p\}\) is an answer set of \(\Pi\) under the GL-semantics, which is also an answer set of the normal derivative \(P_1(\Pi)\) under the GL\(_{nlp}\)-semantics.

However, Hitzler and Seda (1999) raised an open question of characterizing normal derivatives whose answer sets are answer sets of \(\Pi\). Specifically, the problem states that for any interpretation \(I\), determine some normal derivatives such that \(I\) is an answer set of \(\Pi\) under the GL-semantics if and only if \(I\) is an answer set of these normal derivatives under the GL\(_{nlp}\)-semantics.

It is particularly interesting to note that our characterization of the GL-semantics using the disjunctive program reduct (see Theorem 2) leads to a satisfactory solution for this open problem. For a logic program \(\Pi\), an interpretation \(I\) and a head selection \(sel\) on \(I\), we denote

\[
P_{sel}(\Pi, I) = \{sel(head(r), I) \leftarrow body(r) \mid r \in ground(\Pi)\}
\]

and

\[
ND(\Pi, I) = \{P_{sel}(\Pi, I) \mid sel\ is\ a\ head\ selection\ on\ I\}.
\]

Note that \(ND(\Pi, I)\) is the collection of normal derivatives obtained by applying every head selection on \(I\). Thus for any head selection \(sel\) on a model \(I\), we have

\[
P_{sel}(\Pi, I) = \{sel(head(r), I) \leftarrow body(r) \mid r \in ground(\Pi)\}
\]

= \(\Pi_{sel}^I \cup \{sel(head(r), I) \leftarrow body(r) \mid r \in ground(\Pi)\ and\ body(r)\ is\ not\ satisfied\ by\ I\}\)

Then the following theorem gives a solution to the above open problem.

**Theorem 6** Let \(\Pi\) be a simple disjunctive program and \(I\) an interpretation. Then \(I\) is an answer set of \(\Pi\) under the GL-semantics if and only if \(I\) is an answer set of every normal derivative in \(ND(\Pi, I)\) under the GL\(_{nlp}\)-semantics.

The proof of this theorem requires the following lemma.

**Lemma 3** Let \((\Pi_{sel}^I)^I\) and \((P_{sel}(\Pi, I))^I\) be the GL-reduct of \(\Pi_{sel}^I\) and \(P_{sel}(\Pi, I)\), respectively. Then an interpretation \(I\) is a minimal model of \((\Pi_{sel}^I)^I\) if and only if \(I\) is a minimal model of \((P_{sel}(\Pi, I))^I\).

For example, the above logic program \(\Pi = \{p \mid q \leftarrow \neg s\}\) has three models: \(I_1 = \{p\}, I_2 = \{q\}\) and \(I_3 = \{p, q\}\). For \(I_1, ND(\Pi, I_1)\) has only one normal derivative \(P_{sel}(\Pi, I_1) = \{p \leftarrow \neg s\}\), where \(sel\) selects \(p\) from the rule head \(p \mid q\). \(I_1\) is an answer set of \(P_{sel}(\Pi, I_1)\) under the GL\(_{nlp}\)-semantics, and by Theorem 6 is an answer set of \(\Pi\) under the GL-semantics. Similarly, \(I_2\) is an answer set of \(\Pi\). For \(I_3, ND(\Pi, I_3)\) consists of two normal derivatives, \(P_{sel}(\Pi, I_3) = \{p \leftarrow \neg s\}\) and \(P_{sel}(\Pi, I_3) = \{q \leftarrow \neg s\}\), where \(sel_1\) and \(sel_2\) select \(p\) and \(q\) from the rule head \(p \mid q\), respectively. \(I_3\) is not an answer set of \(P_{sel}(\Pi, I_3)\), and by Theorem 6 is not an answer set of \(\Pi\).

Finally, we note that the normal derivatives of a simple disjunctive program \(\Pi\) coincide with the split programs of Sakama and Inoue (1993), whose answer sets constitute the so called “possible worlds” of
Π. In the preliminary work, Sakama (1989) had defined possible worlds of a positive simple disjunctive program Π as the minimal models of all its split programs resp. derivatives $P(Π)$, which coincide with the “possible models” in the semantics of Chan (1993) and the sustained models considered by Decker and Casamayor (1994), which used a level mapping on the atoms to foster derivability from facts similar as Fages (1994). Apparently, the possible worlds semantics is more liberal than both the GL-semantics and the DI-semantics; regarding answer set computation, it is for simple disjunctive programs “cheaper,” as computing some possible world is feasible in nondeterministic polynomial time, while computing some DI- resp. GL-answer set is $\Delta^p_2[\log n]$-hard resp. $\Pi^p_2$-hard.

8 Conclusion

In this paper we reconsider answer set semantics for disjunctive programs. We observed that due to its unique rule head atom requirement the GL-semantics is a bit too restrictive and may exclude some desired answer sets. We then presented a novel alternative, i.e., the DI-semantics, which overcomes the limitations of the GL-semantics. The new semantics is general and applicable to extend any answer set semantics $\mathcal{X}$ for normal programs to disjunctive programs.

By replacing the base semantics $\mathcal{X}$ in the general DI-semantics with the GL$_{nlp}$-semantics, we induced a DI-semantics for simple disjunctive programs. We applied it to formalize a Generalized Strategic Companies problem and a relaxed version of the Russell’s Barber paradox. We also presented a novel characterization of the GL-semantics in terms of a disjunctive program reduct. This characterization leads us to giving a satisfactory solution to the open problem presented by Hitzler and Seda (1999) about characterizing split normal derivatives of a simple disjunctive program Π such that answer sets of the normal derivatives are answer sets of Π under the GL-semantics.

By replacing $\mathcal{X}$ in the general DI-semantics with the well-justified semantics, we induced a DI-semantics for general disjunctive programs. This closes an open issue presented in (Shen et al., 2014) about extending the well-justified semantics from general normal programs to general disjunctive programs.

We showed that deciding whether a propositional simple disjunctive program Π has DI-answer sets is NP-complete, and deciding whether a ground literal is true in some (resp. every) DI-answer set of Π is $\Sigma^p_2$-complete (resp. $\Pi^p_2$-complete). This shows an additional advantage over the GL-semantics, as deciding whether Π has GL-answer sets is $\Sigma^p_2$-complete. For propositional general disjunctive programs, the complexity increases by one level in the polynomial hierarchy.

As ongoing work, we are considering an implementation of the DI-semantics for simple disjunctive programs by updating the existing ASP reasoner DLVHEX (http://www.kr.tuwien.ac.at/research/systems/dlvhex).

References


Appendix: Proofs

Proof of Theorem 1  For any GL-answer set $I$ of $\Pi$, $I$ is both a minimal model of $\Pi$ and a minimal model of the GL-reduct $\Pi^I$. For any head selection $sel$, let $(\Pi^I_{sel})^I$ be the GL-reduct of $\Pi^I_{sel}$.

We first show the case that $\Pi$ is a simple normal program. Then for any interpretation there is a unique selection function $sel$. For a model $I$ we also have $(\Pi^I_{sel})^I = \Pi^I$.

($\Rightarrow$) Let $I$ be an DI-answer set of $\Pi$. Then $I$ is an answer set of $\Pi^I_{sel}$ under the GLnlp-semantics, i.e., $I$ is a minimal model of $(\Pi^I_{sel})^I$ and thus is a minimal model of $\Pi^I$. Hence $I$ is an answer set of $\Pi$ under the GL-semantics.

($\Leftarrow$) Let $I$ be an answer set of $\Pi$ under the GL-semantics. Then $I$ is a minimal model of $\Pi^I$ and thus is a minimal model of $(\Pi^I_{sel})^I$. This means that $I$ is an answer set of $\Pi^I_{sel}$ under the GLnlp-semantics. As $I$ is a minimal model of $\Pi$, $I$ is a DI-answer set of $\Pi$.

Next we show the case that $\Pi$ is a positive simple disjunctive program. Note that for any $I$ and $sel$, $(\Pi^I_{sel})^I = \Pi^I$. For a model $I$, $A_1 \mid \cdots \mid A_k \leftarrow B_1 \land \cdots \land B_m$ is a rule in $\Pi^I$ whose body is satisfied by $I$ if for $sel(A_1 \mid \cdots \mid A_k, I) = A_i$, $A_i \leftarrow B_1 \land \cdots \land B_m$ is a rule in $\Pi^I_{sel}$ and thus in $(\Pi^I_{sel})^I$.

($\Rightarrow$) Let $I$ be a DI-answer set of $\Pi$. Then, (1) for some head selection function $sel$, $I$ is an answer set of $\Pi^I_{sel}$ under the GLnlp-semantics, i.e., $I$ is a minimal model of $\Pi^I_{sel}$, and (2) $\Pi$ has no model $J \subset I$ satisfying condition (1). The condition (2) means that for any model $J \subset I$ of $\Pi$ and any head selection function $sel$, $J$ is not a minimal model of $\Pi^I_{sel}$.

Assume towards the contradiction that $I$ is not a minimal model of $\Pi$. Let $J \subset I$ be a minimal model of $\Pi$. Then (1) $J$ is a minimal model of $\Pi^I$, and (2) for any head selection function $sel$, $J$ is not a minimal model of $\Pi^I_{sel}$. Let $sel$ be an arbitrary head selection function and $L \subset J$ be a minimal model of $\Pi^I_{sel}$. For any rule $r$ in $\Pi^I$ of the form $A_1 \mid \cdots \mid A_k \leftarrow B_1 \land \cdots \land B_m$ such that $L$ satisfies $body(r)$ (and thus $J$ satisfies $body(r)$), there must be a rule $r'$ in $\Pi^I_{sel}$ of the form $A_i \leftarrow B_1 \land \cdots \land B_m$, where $sel(A_1 \mid \cdots \mid A_k, J) = A_i$, such that $L$ satisfies $A_i$ and thus satisfies $head(r)$. This means that $L$ is a model of $\Pi^I$, contradicting the assumption that $J$ is a minimal model of $\Pi^I$. Hence $I$ is a minimal model of $\Pi$ and thus an answer set under the GL-semantics.
(⇐⇒) Let $I$ be an answer set of $\Pi$ under the GL-semantics. Then $I$ is both a minimal model of $\Pi$ and $\Pi^I$. Assume towards a contradiction that for any head selection $sel$, $I$ is not an answer set of $\Pi^I_{sel}$ under the GL$_{nlp}$-semantics, i.e., $I$ is not a minimal model of $\Pi^I_{sel}$. Let $J \subset I$ be a minimal model of $\Pi^I_{sel}$. For any rule $r$ in $\Pi^I$ of the form $A_1 | \cdots | A_k \leftarrow B_1 \land \cdots \land B_m$ such that $J$ satisfies $body(r)$ (and thus $I$ satisfies $body(r)$), there must be a rule in $\Pi^I_{sel}$ of the form $A_i \leftarrow B_1 \land \cdots \land B_m$, where $sel(A_1 | \cdots | A_k, I) = A_i$, such that $J$ satisfies $A_i$ and thus satisfies $head(r)$. This means that $J$ is a model of $\Pi^I$, contradicting the assumption that $I$ is a minimal model of $\Pi^I$. Hence for some head selection $sel$, $I$ is an answer set of $\Pi^I_{sel}$ under the GL$_{nlp}$-semantics. As $I$ is a minimal model of $\Pi$, $I$ is a DI-answer set of $\Pi$.

Proof of Theorem 2 (⇐⇒) For any GL-answer set $I$ of $\Pi$, $I$ is both a minimal model of $\Pi$ and a minimal model of the GL-reduct $\Pi^I$. For any head selection $sel$, let $(\Pi^I_{sel})^I$ be the GL-reduct of $\Pi^I_{sel}$. Then, $A_1 | \cdots | A_k \leftarrow B_1 \land \cdots \land B_m \land \neg C_1 \land \cdots \land \neg C_n$ is a rule in $ground(\Pi)$ whose rule body is satisfied by $I$ iff $A_1 | \cdots | A_k \leftarrow B_1 \land \cdots \land B_m$ is a rule in $\Pi^I$ iff for $sel(A_1 | \cdots | A_k, I) = A_i$, $A_i \leftarrow B_1 \land \cdots \land B_m \land \neg C_1 \land \cdots \land \neg C_n$ is a rule in $\Pi^I_{sel}$ if $A_i \leftarrow B_1 \land \cdots \land B_m$ is a rule in $(\Pi^I_{sel})^I$.

Let $I$ be a GL-answer set of $\Pi$. Then $I$ is both a minimal model of $\Pi$ and $\Pi^I$. Assume towards a contradiction that for some head selection $sel$, $I$ is not an answer set of $\Pi^I_{sel}$ under the GL$_{nlp}$-semantics. Then $I$ is not a minimal model of $(\Pi^I_{sel})^I$. Let $J \subset I$ be a minimal model of $(\Pi^I_{sel})^I$. For any rule $r$ in $\Pi^I$ of the form $A_1 | \cdots | A_k \leftarrow B_1 \land \cdots \land B_m$ such that $J$ satisfies $body(r)$ (and thus $I$ satisfies $body(r)$), there must be a rule in $(\Pi^I_{sel})^I$ of the form $A_i \leftarrow B_1 \land \cdots \land B_m$, where $sel(A_1 | \cdots | A_k, I) = A_i$, such that $J$ satisfies $A_i$ and thus satisfies $head(r)$. This means that $J$ is a model of $\Pi^I$, contradicting the assumption that $I$ is a minimal model of $\Pi^I$. Hence for every head selection function $sel$, $I$ is an answer set of $\Pi^I_{sel}$ under the GL$_{nlp}$-semantics.

(⇐⇒) Assume that for every head selection $sel$, $I$ is an answer set of $\Pi^I_{sel}$ under the GL$_{nlp}$-semantics, i.e., $I$ is a minimal model of $(\Pi^I_{sel})^I$. Assume towards a contradiction that $I$ is not a GL-answer set of $\Pi$, i.e., $I$ is not a minimal model of $\Pi^I$. Let $J \subset I$ be a minimal model of $\Pi^I$. Consider a head selection $sel$ with the property that for every rule $r$ in $\Pi^I$ of the form $A_1 | \cdots | A_k \leftarrow B_1 \land \cdots \land B_m$ such that $J$ satisfies $body(r)$ (and thus $I$ satisfies $body(r)$), $sel(A_1 | \cdots | A_k, I) = A_i$ with $A_i$ being satisfied by $J$. Such $A_i$ must exist because $J$ is a model of $\Pi^I$. Then $J$ is a model of $(\Pi^I_{sel})^I$. This means that $I$ is not a minimal model of $(\Pi^I_{sel})^I$, contradicting our assumption. Hence, $I$ is a GL-answer set of $\Pi$.

Proof of Theorem 3 The membership parts can be shown with simple guess and check algorithms: a head selection $sel$ and an interpretation $I$ for condition (1) of a DI-WJ answer set can be guessed and checked, by verifying that $I$ is a GL-answer set of $\Pi^I_{sel}$, the latter is feasible in polynomial time (as the program $\Pi^I_{sel}$ is normal). If this succeeds (which is in NP), then some DI-WJ answer set exists, which proves (i). For (ii) and (iii), we require that $I$ does not satisfy $L$ resp. satisfies $L$ and check that no $J \subset I$ exists that satisfies condition (1) for some $sel'$; the latter check is in co-NP. In conclusion, we obtain membership (i) in NP, (ii) in $\Pi^p$, and (iii) in $\Sigma^p_2$, respectively.

The matching hardness results are immediate by Theorem 1 and the facts that GL-answer set existence for a simple normal program is NP-complete, cf. (Gelfond and Lifschitz, 1991), and that cautious and brave inference of a literal from the GL-answer sets of a simple positive disjunctive program is $\Pi^p$- resp. $\Sigma^p_2$-hard, as shown in (Eiter and Gottlob, 1995).

Proof of Lemma 1 Indeed, if condition (1) holds, then there exists some head selection function $sel$ on $I$ such that for every atom $A \in I$ it holds that $lfp(T_{\Pi}(\emptyset, I, sel)) \cup \neg I^- \models A$. To witness this, we can guess $sel$, compute $lfp(T_{\Pi}(\emptyset, I, sel))$ and then do all the entailment checks in polynomial time using an NPoracle, which implies membership in $\Sigma^p_2$. Note that if $I$ is a simple disjunctive program, then computing
Informally, every assignment $\sigma$ to $X = X^1X^2X^3$ in which every $X^j_i$ and $\bar{X}^j_i$ have opposite values and all variables in $X^2X^3$ are true will be a model of $\Psi(X)$, and is a candidate for a minimal model. In fact, $\sigma$ is not minimal if and only if there exists some assignment $\sigma'$ that (a) coincides with $\sigma$ on $X^1X^1$, (b) assigns false to some variable in $X^2$, and (c) is such that $\forall X^3 \neg F(\sigma(X), X^3)$ evaluates to true. That is, $\sigma$ is minimal if and only if the QBF $\forall X^2 \exists X^3 F(\sigma(X^1), X^2, X^3)$ evaluates to true. Note that in every such $\sigma$, the variable $X^2_0$ is true, and that in every minimal model $\sigma'$ smaller than $\sigma$ we must have that $X^2_0$ is false. Hence, $A = X^2_0$ is true in some minimal model of $\Psi(X)$ if and only if the QBF $\Phi$ is true. This proves the $\Sigma_3^P$-hardness.

Prove of Theorem 4 Membership in $\Sigma_2^P$ holds since we can guess a DI-WJ answer set $I$ of $\Pi$ and check it by Lemma 1 using a $\Sigma_2^P$ oracle call. The guess and oracle check can be combined into a single (polynomial size) guess followed by a call to a co-NP oracle; this shows membership in $\Sigma_2^P$. The $\Sigma_2^P$-hardness is inherited from the $\Sigma_2^P$-completeness of deciding answer set existence under the well-justified answer set semantics of general normal programs (Shen et al., 2014), which by Corollary 2 coincides with the well-justified answer set semantics.

Proof of Proposition 1 By Lemma 1, we can test using a $\Sigma_2^P$ oracle that condition (1) of Definition 9 holds for $I$. Furthermore, we can test using a $\Pi_2^P$ oracle that no smaller $J \subset I$ exists that satisfies condition (1); that is, that for every $J \subset I$ and selection function $sel'$, it holds that (*) some atom $A \in J$ exists such that $lfp(T_{\Pi}(\emptyset, J, sel')) \cup \neg J \not\models A$. To witness (*) for $sel'$ and $J$, we can guess (a) for each $i = 1, \ldots, m$, where $m = |I|$, a set $N_i \subseteq I$ of rules $r$ with associated interpretations $J_i^r$ such that

$$J_i^r \text{ satisfies the formulas } S_{i-1} \cup \neg J \cup \{\neg \text{body}(r)\}$$

(6)

where $S_0 = \emptyset$ and $S_i = \{sel'(\text{head}(r), J) \mid r \in I \setminus N_i\}$, and (b) an interpretation $J'$ such that

$$J' \text{ satisfies } S_m \cup \neg J \cup \{\neg A\}.$$  

(7)

Informally, $N_i$ is a subset of those rules $r$ in $I$ whose body is not entailed by $T_{\Pi}^{i-1}(\emptyset, J, sel') \cup \neg J$, witnessed by the interpretation $J_i^r$; consequently, $S_i$ is a superset of $T_{\Pi}^{i}(\emptyset, J, sel')$. Hence if (6) holds for all $j = 1, \ldots, m$ then $S_m$ is a superset of $lfp(T_{\Pi}(\emptyset, J, sel'))$, and if in addition (7) holds, then we have $lfp(T_{\Pi}(\emptyset, J, sel')) \cup \neg J \not\models A$. Conversely, if $lfp(T_{\Pi}(\emptyset, J, sel')) \cup \neg J \not\models A$ then by guessing all $N_i$, all $J_i^r$ and $J'$ appropriately (6) and (7) will hold. As the guess has polynomial size, it follows that deciding (*) is in NP; consequently, to decide that for all $J \subset I$ and $sel'$ condition (*) holds is in $\Pi_2^P$, i.e., deciding whether condition (1) does not hold for some $J \subset I$ is in $\Pi_2^P$. Thus as the problem amounts to the conjunction of a $\Sigma_2^P$ and a $\Pi_2^P$ problem, it is in $D_2^P$.

Proof of Lemma 2 Membership in $\Sigma_2^P$ is immediate by a guess and check argument, using a $\Pi_2^P$ oracle.

As for the hardness part, let $\Phi = \exists X^1 \forall X^2 \exists X^3 F(X^1, X^2, X^3)$ be a QBF with variables $X^i = \{X_i^1, \ldots, X_i^{n_i}\}$, $i = 1, 2, 3$. Let $\bar{X}^1 = \{\bar{X}_1^1, \ldots, \bar{X}_1^{n_1}\}$ be a copy of $X^1$, and let $X^2_0$ be a fresh variable. Define

$$\Psi(X^1 \bar{X}^1 X^2 X^3_0) = \forall X^3(\bigwedge_{i=1}^{n_1}(X_i^1 \equiv \bar{X}_i^1) \land (\bigwedge_{j=0}^{n_2}(X_j^2 \lor \neg F(X^1, X^2, X^3)))].$$

Informally, every assignment $\sigma$ to $X = X^1X^1X^2X^3_0$ in which every $X^j_i$ and $\bar{X}^j_i$ have opposite values and all variables in $X^2X^3_0$ are true will be a model of $\Psi(X)$, and is a candidate for a minimal model. In fact, $\sigma$ is not minimal if and only if there exists some assignment $\sigma'$ that (a) coincides with $\sigma$ on $X^1X^1$, (b) assigns false to some variable in $X^2$, and (c) is such that $\forall X^3 \neg F(\sigma(X), X^3)$ evaluates to true. That is, $\sigma$ is minimal if and only if the QBF $\forall X^2 \exists X^3 F(\sigma(X^1), X^2, X^3)$ evaluates to true. Note that in every such $\sigma$, the variable $X^2_0$ is true, and that in every minimal model $\sigma'$ smaller than $\sigma$ we must have that $X^2_0$ is false. Hence, $A = X^2_0$ is true in some minimal model of $\Psi(X)$ if and only if the QBF $\Phi$ is true. This proves the $\Sigma_3^P$-hardness.
Proof of Theorem 5 As for the membership parts, we can guess a Di-WJ answer set \( I \) of \( \Pi \) such that \( I \) satisfies \( L \) (resp. does not satisfy \( L \)) and verify \( I \) with the help of a \( \Sigma^P_2 \) oracle in polynomial time as follows from Proposition 1.

The \( \Sigma^P_2 \) hardness of brave reasoning is shown by the following reduction from MINQASAT. Let \( \Phi = \forall Y. E(X, Y) \) be a QBF as in Lemma 2, where \( X = \{X_1, \ldots, X_n\} \) and \( Y = \{Y_1, \ldots, Y_m\} \). We construct the following logic program \( \Pi \):

\[
\begin{align*}
X_i \mid \neg X_i & \leftarrow \quad \text{for all } i = 1, \ldots, n \\
Y \land p & \leftarrow E(X, Y) \\
p & \leftarrow \neg p
\end{align*}
\]

where \( p \) is a fresh atom and \( Y \) stands for \( Y_1 \land \cdots \land Y_m \). Intuitively, the first rules guess a truth assignment \( \sigma \) to the variables in \( X \), and the second rule checks whether under this assignment \( E(X, Y) \) is a tautology, i.e., \( \forall Y. E(\sigma(X), Y) \) evaluates to true; in this case, the atom \( p \) is derived, which must be true in every DI-WJ answer set of \( \Pi \). Likewise all \( Y_i \) must be true, and hence no \( \neg Y_j \) occurs in \( \neg I^- \). In fact, the truth assignments \( \sigma \) to \( X \) such that \( \forall Y. (\sigma(X), Y) \) evaluates to true correspond one-to-one to the models \( I = I_\sigma \) of \( \Pi \) that fulfill condition (1) of Definition 8, where \( I_\sigma = \{X_i \mid \sigma(X_i) = 1\} \cup Y \cup \{p\} \). Consequently, by condition (2) of this definition, the DI-WJ answer sets of \( \Pi \) correspond to the minimal assignments \( \sigma \) to \( X \) such that \( \forall Y. E(\sigma(X), Y) \) evaluates to true. Hence, \( \Phi \) with \( A \in X \) is a yes-instance of MINQASAT if and only if \( A \) is true in some DI-WJ answer set of \( \Pi \). This proves \( \Sigma^P_3 \)-hardness of brave reasoning. The \( \Pi^P_3 \)-hardness of cautious reasoning results by asking whether \( \neg A \) is true in all DI-WJ answer sets of \( \Pi \).

Remarks. (1) We can extend the construction to show that the minimal models of a QBF \( \exists Z \forall Y. E(X, Y, Z) \) correspond to the DI-WJ answer sets of a program \( \Pi \) constructed in polynomial time. To this aim, we add rules

\[
Z_i \mid \hat{Z}_i \leftarrow \quad \text{for all } Z_i \in Z
\]

and change the rule \( Y \land p \leftarrow E(X, Y) \) to

\[
Z \land \hat{Z} \land Y \land p \leftarrow \hat{E}(X, Y, Z) \lor \bigvee_{i=1}^{k} (Z_i \land \hat{Z}_i)
\]

where \( Z \) (resp. \( \hat{Z} \)) stands for \( \bigwedge_{Z_i \in Z} Z_i \) (resp. \( \bigwedge_{Z_i \in Z} \hat{Z}_i \)), and \( \hat{E}(X, Y, Z) \) stands for \( E(X, Y, Z) \) put in negation normal form, where each occurrence of \( \neg Z_i \) is replaced by \( \hat{Z}_i \). Intuitively, any DI-WJ answer set must contain each \( Z_i \) and the fresh \( \hat{Z}_i \), which simulates the complement of \( Z_i \). When it comes to applying the rule with \( \hat{E}(X, Y, Z) \) in the body in the fixpoint construction, one of \( Z_i \) or \( \hat{Z}_i \) is selected and informally set to true, in addition to the selected literals \( X_i \), resp. \( \neg X_i \). We can then derive \( p \) iff \( E(X, Y, Z) \) is under the assignment \( \sigma \) of \( X \) and \( Z \) a tautology. Thus, the DI-WJ answer sets of \( \Pi \) are given by extending \( I_\sigma \) from above with \( Z \) and \( \hat{Z} \), i.e., by \( \{X_i \mid \sigma(X_i) = 1\} \cup Y \cup Z \cup \hat{Z} \cup \{p\} \) where \( \exists Z \forall Y. E(\sigma(X), Y, Z) \) evaluates to true.

(2) From an extended construction, we can conclude \( D^P_2 \)-hardness of DI-WJ answer set checking. First assume that \( E(X, Y) \) is satisfied if all variables in \( X \) are set to 1, i.e., \( \forall Y. E(X = 1, Y) \) evaluates to true. Then \( I_1 = X \cup Y \cup \{p\} \) satisfies condition (1) and is a WI-DJ answer set of the program, denoted \( \Pi_1 \), iff for every assignment \( \sigma \) that does not set all \( X_i \) to 1, we have that \( \neg E(\sigma(X), Y) \) is satisfiable, i.e., \( \Phi_1 = \exists X \neq 1 \forall Y. E(X, Y) \) evaluates to false; deciding this is \( \Pi^P_2 \)-hard. Second, if we use the extended construction for QBFs \( \exists Z \forall Y. E(X, Y, Z) \) from Remark (1) and set \( X \) void, then \( I_2 = Y \cup Z \cup \hat{Z} \cup \{p\} \) is a WI-DJ answer set of the program, denoted \( \Pi_2 \), iff \( \Phi_2 = \exists Z \forall Y. E(Y, Z) \) evaluates to true. It follows that
$I = I_1 \cup I_2$ is a DI-WJ answer set of the program $\Pi = \Pi_1 \cup \Pi_2$ iff the QBFs $\Phi_1$ and $\Phi_2$ evaluate to false and true, respectively; hence, DI-WJ answer set checking $D^p_{\text{WJ}}$-hard.

(3) The above reduction works also for FLP answer sets as the base semantics: the formula $E(X, Y)$ can not be derived from the literals $X_i$ resp. $\neg X_i$ chosen by $sel$ for an assignment $\sigma$ in the least fixpoint construction iff $I_\sigma$ is not a minimal model of the reduct $\Pi_{sel}$.

Proof of Lemma 3  
As

$P_{sel}(\Pi, I) = \Pi_{sel} \cup \{sel(head(r), I) \leftarrow body(r) \mid r \in \text{ground}(\Pi) \text{ and } body(r) \text{ is not satisfied by } I\}$

$(P_{sel}(\Pi, I))^I$ consists of $(\Pi_{sel})^I$ plus some rules of the form $sel(head(r), I) \leftarrow B_1 \land \cdots \land B_m$, where every $B_i$ is a ground atom and some $B_i$ is not in $I$ (thus the rule body is not satisfied by $I$). Therefore, for any $J \subset I$, for every rule $r$ in $(P_{sel}(\Pi, I))^I \setminus (\Pi_{sel})^I$, $J$ does not satisfy $body(r)$.

$(\implies)$ Let $I$ be a minimal model of $(\Pi_{sel})^I$ and assume towards a contradiction that $I$ is not a minimal model of $(P_{sel}(\Pi, I))^I$. Let $J \subset I$ be a minimal model of $(P_{sel}(\Pi, I))^I$. Then $J$ is also a model of $(\Pi_{sel})^I$ since $(\Pi_{sel})^I \subseteq (P_{sel}(\Pi, I))^I$, contradicting that $I$ is a minimal model of $(\Pi_{sel})^I$.

$(\impliedby)$ Let $I$ be a minimal model of $(P_{sel}(\Pi, I))^I$. Then $I$ is a model of $(\Pi_{sel})^I$. Assume towards a contradiction that $I$ is not a minimal model of $(P_{sel}(\Pi, I))^I$. Let $J \subset I$ be a minimal model of $(P_{sel}(\Pi, I))^I$. As mentioned above, for every rule $r$ in $(P_{sel}(\Pi, I))^I \setminus (\Pi_{sel})^I$, $J$ does not satisfy $body(r)$. This means that $J$ is also a model of $(P_{sel}(\Pi, I))^I$, contradicting that $I$ is a minimal model of $(P_{sel}(\Pi, I))^I$.  

Proof of Theorem 6  
$I$ is a GL-answer set of $\Pi$ iff by Theorem 2, for every head selection function $sel$, $I$ is an answer set of $\Pi_{sel}$ under the GL_{nlp}-semantics iff for every head selection function $sel$, $I$ is a minimal model of $(\Pi_{sel})^I$ iff for every head selection function $sel$, by Lemma 3 $I$ is a minimal model of $(P_{sel}(\Pi, I))^I$ iff $I$ is an answer set of every normal derivative in $ND(\Pi, I)$ under the GL_{nlp}-semantics.  

More Information on Function Complexity

The class $\text{FP}^{\Sigma^p_2}$ consists of all multi-valued total functions $f$ such that for every input $x$ some value $y \in f(x)$ is computable by a deterministic Turing machine in polynomial time with at most $O(\log |x|)$ many queries to a witness oracle $O$ in $\Sigma^p_1$ (Buss et al., 1993). The witness oracle informally decides a problem in $\Sigma^p_1$, but provides for an oracle query $x_O$ in case the answer is “yes” in addition a string $y_O$ of size polynomial in $|x_O|$ (a “witness” for the answer) that can be checked with a single call to a $\Pi^p_{i-1}$ oracle; e.g., for a SAT oracle ($i = 1$), a natural witness would be a satisfying assignment to the input formula. Notably, different witnesses may lead to different output of the machine, which is given by the content of a designated output tape at the end of each run.\footnote{Importantly, the machine always accepts and produces some output; otherwise $\Sigma^p_2$-hard problems, e.g. finding models of QBF $\forall Y . E(X,Y)$, could be solved in $\text{FP}^{\Sigma^p_2}$-complete for functions $f$ in $\text{FP}^{\log \cdot \text{wit}}$ but “only” $\text{FNP}^{\log \cdot n}$-complete for functions $f$ in $\text{FNP}^{\log \cdot \text{optP}}$; this is due to the possibly exponentially many witnesses for a query answer.}

Chen and Toda (1995) introduced the class $\text{FNP}^{\log \cdot \text{wit}}$, which consists of the partial multi-valued functions that can be solved by a non-deterministic Turing machine in polynomial time with logsize oracle advice $a(x)$ from an NP-optimization problem. More precisely, for an input $x$ and $a(x)$, some value $y \in f(x)$ is nondeterministically computed by the machine, where $a(x)$ is the binary presentation of the maximum output of a metric Turing machine, i.e., a nondeterministic Turing machine that outputs an integer in each branch, on input $x$, where each integer is polynomial in the size of $x$ (thus $|a(x)| = O(\log |x|)$)
holds); for example, $a(x)$ may be the maximum size of a clique in a graph, or the maximum size of a model for a SAT instance. It can be shown that $\text{FNP}/\text{OptP}[\log n]$ and $\text{FNP}[\log, \text{wit}]$ coincide on total multi-valued functions.\footnote{Technically, $\text{FNP}[\log, \text{wit}]$ requires that each input has some output. However, each partial function $f$ in $\text{FNP}/\text{OptP}[\log n]$ can be turned into a total function $f_C$ such that $f_C(x) = f(x)$ if $f(x)$ is defined and $f_C(x) = \nabla$ otherwise, where $\nabla$ is a fresh symbol; this function $f_C$ is computable in $\text{FNP}[\log, \text{wit}]$.}

Chen and Toda proved that computing some maximal model of a QBF $\exists Z.E(X, Z)$ ($= \text{X-MAX-MODEL}$) is complete for $\text{FNP}/\text{OptP}[\log n]$, under the following polynomial-time reduction for (partial) multi-valued functions: $f_1$ reduces to $f_2$, if there are functions $g_1, g_2$ computable in polynomial time such that for every $x$, (a) $g_1(x)$ is an instance of $f_2$ and has some output, i.e., $f_2(g_1(x)) \neq \emptyset$, iff $f_1(x)$ has some output, and (b) every $y \in f_2(g_1(x))$ gives rise to some output $g_2(x, y) \in f_1(x)$. Note that without loss of generality, one can assume that $\exists Z.E(X, Z)$ has some model. In this setting, the problem is then also complete for $\text{FNP}[\log, \text{wit}]$.

To provide more detail, membership of computing some maximal model of a QBF $\exists Z.E(X, Z)$ is in $\text{FNP}[\log, \text{wit}]$ as established follows: we do a binary search on $k \geq 0$, whether $\exists Z.E(X, Z)$ has some model $\sigma$ of size at least $k$; to this end, $\exists Z.E(X, Z)$ and $k$ are transformed into a respective SAT instance $F_k$ that we ship to a SAT oracle. When we know the maximum size $k^*$, we can again ask the oracle for a witness of $F_{k^*}$, which is then the result. The membership of computing some maximal model of a QBF $\exists Z E(X, Y, Z) \in \text{FP}^{\log_2}[\log, \text{wit}]$ can be established analogously, using a $\Sigma_0^p$ oracle.

The $\text{FP}^{\log, \text{wit}}$-hardness of computing some maximal model of a QBF $\exists Z.E(X, Z)$ an be established, using Chen and Toda's key ideas by a generic Turing machine encoding as follows. A possible run of the machine $M$ with witness oracle $O$ in NP on input $x$, relative to hypothetical results $a = a_1 a_2 \cdots a_{\log |x|}$ of the answer $a_i \in \{\text{"yes", \"no\"}\}$ to the $i$-th query $x_i$, can be encoded to a SAT instance $E(M, x, a)$, where for $a_i = \text{"yes"}$ a proper witness $y_i$ for $x_i$ is characterized as a SAT instance $E_{x_i}$ (this is possible by Cook's Theorem). If we view $a$ as a number in binary, where \"yes\"=1 and \"no\"=0, then any model of $E(M, x, a)$ where $a$ is maximum reflects a proper run of $M$ on $x$, as no $a_i = \text{\"no\"}$ can be flipped to $a_i = \text{\"yes\"}$ to obtain a larger value.\footnote{Note that here it is important that each run of the machine produces some output.}

We couch this now into computing some maximal model of a QBF $\exists Y.E(X, Y)$ by introducing variables $X = \{X_1, \ldots, X_n\}$ where $n = 2^{\log |x|}$ and expressing in $E$ that for some $i$, $X_1, \ldots, X_i$ must be true and $X_{i+1}, \ldots, X_n$ must be false; intuitively, $i$ expresses a number $a^{(i)} = a_1^{(i)} \cdots a_{\log |x|}^{(i)}$ in tally notation. In $E$ we express that if $X_i \land \neg X_{i+1}$ is true, then $a = a^{(i)}$ holds; the rest of $E$ contains $E(M, x, a)$, and we let $Y$ be all variables not in $X$. The maximal models of $\exists Y.E(X, Y)$ then correspond to proper runs of $M$ on $x$ (but not every proper run might be covered). However, at this point we miss the output $y$ computed by $M$ on $x$. In order to bring it in, we can duplicate each $X_j$ to a block $X_{j,1}, \ldots, X_{j,m}$ of variables, where $m$ is large enough, and request that they are all true if $j < i$, all false if $j > i$ and for $j = i$ encode $y$ (in binary). Each maximal model of the adapted QBF $\exists Y.E(X, Y)$ encodes then a proper run of $M$ on $x$ with some output $y$. In order to get all outputs for the answers $a(x)$, we may add the set complement $\overline{y}$ to $X$; this will ensure that different outputs $y$ will be incomparable under model maximization.

Notably, the restriction of $\text{X-MAX-MODEL}$ to cardinality-maximal (i.e., largest size) models of a QBF $\exists Z.E(X, Z)$ is also complete for $\text{FNP}[\log, \text{wit}]$ and $\text{FNP}/\text{OptP}[\log n]$; the hardness part follows by the complement addition $\overline{y}$, as then cardinality- and subset-maximal models coincide. In fact, the restriction is hard even in absence of $\exists Z$, i.e., for SAT instances $E(X)$; this can be shown by eliminating $\exists Z$ from $\exists Z.E(X, Z)$ with further variable duplication, such that the block for each variable in $X$ is larger than $Z$.\footnote{Whether subset-minimal $\text{X-MAX-MODEL}$ for $E(X)$ is $\text{FNP}[\log, \text{wit}]-hard$ is unknown.}
The $\text{FP}^{\Sigma_2^p[\log, \text{wit}]}$-hardness of computing some maximal model of a QBF $\exists Z \forall Y. E(X, Y, Z)$ can be shown similarly, where the witness oracle calls are modeled as QBFs $\exists X_i \forall Y_i. E(X_i, Y_i)$ and the resulting formula is transformed into the desired form.

Computing minimal models of a QBF $\exists Z. E(X, Z)$ (resp., $\exists Z \forall Y. E(X, Y, Z)$) is trivially polynomial-time equivalent to computing maximal models of such a QBF.

It thus follows from Remark (1) of proof of Theorem 5 that computing some DI-WJ answer set of a general disjunctive program $\Pi$ is $\text{FP}^{\Sigma_2^p[\log, \text{wit}]}$-hard. The construction there can be adapted for QBFs $\exists Z. E(X, Z)$, where $E(X, Z)$ is in CNF, to simple disjunctive programs, which establishes hardness for $\text{FP}^{\text{NP}[\log, \text{wit}]}$. 