First-Order Expressibility and Boundedness of Disjunctive Logic Programs

Heng Zhang and **Yan Zhang**
Artificial Intelligence Research Group
School of Computing, Engineering and Mathematics
University of Western Sydney, Australia

Abstract

In this paper, the fixed point semantics developed in [Lobo *et al.*, 1992] is generalized to disjunctive logic programs with default negation and over arbitrary structures, and proved to coincide with the stable model semantics. By using the tool of ultraproducts, a preservation theorem, which asserts that a disjunctive logic program without default negation is bounded with respect to the proposed semantics if and only if it has a first-order equivalent, is then obtained. For the disjunctive logic programs with default negation, a sufficient condition assuring the first-order expressibility is also proposed.

1 Introduction

Disjunctive logic programming enhances the traditional logic programming in both the expressive power and the ability to represent indefinite knowledge. It has been widely recognized as a powerful formalism for database querying [Eiter *et al.*, 1997], knowledge representation and declarative problem solving [Baral, 2003]. The dominating semantics for it in these areas is the stable model semantics [Gelfond and Lifschitz, 1988; Ferraris *et al.*, 2011; Lin and Zhou, 2011], or equivalently the minimal model semantics if no default negation is involved [Lobo *et al.*, 1992].

However, a main drawback of this formalism is its high undecidability or intractability [Chomicki and Subrahmanian, 1990; Eiter and Gottlob, 1997; Eiter *et al.*, 1997]. Therefore, one of the central issues of this formalism is to find tractable or manageable fragments. Identifying first-order expressible fragments is a rather natural way to achieve this goal. In this way, we can then reduce the computations for disjunctive logic programs to the solvers in classical logic, for example, first-order theorem provers for Prolog-style inference, and SAT or SMT solvers for answer-set solving on finite domains.

Recently, a lot of works have been devoted to this task. For example, the first-order expressibility via a single sentence without and with auxiliary predicates were studied by [Cabalar *et al.*, 2009; Bartholomew and Lee, 2010; Zhang and Ying, 2010; Zhang and Zhou, 2010; Chen *et al.*, 2010; 2011; Ferraris *et al.*, 2011; Lee and Meng, 2011; Lifschitz and Yang, 2012] and [Asuncion *et al.*, 2012a; 2012b] respectively, and the first-order expressibility via a possibly infinite theory on finite structures was studied by [Chen *et al.*, 2006]. However, almost all of these works were focusing on only sufficient or only necessary conditions for the first-order expressibility. Unlike them, we are interested in finding conditions that exactly capture the three kinds of first-order expressibility mentioned above. To assure these conditions are useful in identifying first-order expressible fragments, we hope it is intuitively easy to verify whether or not they are satisfied.

The candidate that we choose for such conditions, the boundedness, originally comes from the area of mathematical logic [Moschovakis, 1974; Barwise and Moschovakis, 1978], and has then been thoroughly studied by the community of Datalog as a valuable tool for Datalog optimization [Abiteboul *et al.*, 1995]. An elegant boundedness characterization of the first-order expressibility for Datalog was independently obtained by [Kolaitis and Papadimitriou, 1990] and [Ajtai and Gurevich, 1994]. Therefore, a natural problem is whether or not it can be generalized to the disjunctive case. As the progression on a disjunctive logic program is significantly different from that on a definite one, it is a challenging problem.

In this paper, we first show that the three kinds of first-order expressibility coincide under the stable model semantics. To solve the problem mentioned above, a fixed point semantics, which was proposed in [Lobo *et al.*, 1992] for negation-free disjunctive logic programs over Herbrand structures, is then generalized to disjunctive logic programs with default negation over arbitrary structures. By using the technique of ultraproducts, we prove that a disjunctive logic program without default negation is bounded based on this semantics if and only if it is equivalent to a first-order sentence. For the case with default negation, we also propose a definition for the boundedness and show that it implies the first-order expressibility.

2 Preliminaries

Vocabularies are assumed to be sets of predicate constants and function constants. Every constant is equipped with a natural number, its *arity*. Nullary function constants are also called *individual constants*, and nullary predicate constants are called *proposition constants*. For some technical reasons, a vocabulary is allowed to contain an arbitrary infinite set of proposition constants. Logical symbols are defined as usual, including a countable set of predicate variables and a countable set of individual variables. Predicate constants and variables are simply called *predicates* if no confusion occurs.

Terms, formulae, and sentences of a vocabulary υ (or shortly, υ-terms, υ-formulae, and υ-sentences) are built from υ, equality, and variables in a standard way. The only thing which may be special is that we treat $\neg\varphi$ as a shorthand of $\varphi \rightarrow \bot$, and $\varphi \leftrightarrow \psi$ as the conjunction of $\varphi \rightarrow \psi$ and $\psi \rightarrow \varphi$.

For all formulae φ and theories Σ, let $\upsilon(\varphi)$ and $\upsilon(\Sigma)$ be the sets of all constants occurring in φ and Σ respectively. If $\bar{x} = x_1 \cdots x_m$, τ is the set of X_j for all $1 \leq j \leq n$, x_i and X_j are individual and predicate variables respectively, by $Q\bar{x}$ and $Q\tau$ we denote quantifier blocks $Qx_1 \cdots Qx_m$ and $QX_1 \cdots QX_n$ respectively, where Q is $\forall$ or $\exists$. A first-order (second-order) formula of the form $\forall\bar{x}\vartheta$ ($\forall\tau\vartheta$, respectively) is *universal* if ϑ is first-order (second-order, respectively) quantifier-free.

Every *structure* $\mathbb{A}$ of a vocabulary υ (or shortly, υ-*structure* $\mathbb{A}$) is accompanied by a nonempty set A, the *domain* of $\mathbb{A}$, and interprets each n-ary predicate constant P in υ as an n-ary relation $P^{\mathbb{A}}$ on A, and interprets each n-ary function constant f in υ as an n-ary function $f^{\mathbb{A}}$ on A. A structure is *finite* if its domain is finite; otherwise it is *infinite*. Given two υ-structures $\mathbb{A}, \mathbb{B}$ with the same domain and a set τ of predicates, we write $\mathbb{A} \preccurlyeq_\tau \mathbb{B}$ if the interpretation of each predicate from τ in $\mathbb{A}$ is a subset of that in $\mathbb{B}$, and interpretations of the other predicates and functions in $\mathbb{A}$ are the same as that in $\mathbb{B}$; we write $\mathbb{A} \prec_\tau \mathbb{B}$ if $\mathbb{A} \preccurlyeq_\tau \mathbb{B}$ holds but $\mathbb{B} \preccurlyeq_\tau \mathbb{A}$ does not.

Every *assignment* in a structure $\mathbb{A}$ is a function α that maps each individual variable to an element of A and that maps each predicate variable to a relation on A of the same arity. Given an υ-formula φ and an assignment α in $\mathbb{A}$, we write $(\mathbb{A}, \alpha) \models \varphi$ if α *satisfies* φ in $\mathbb{A}$ in the standard way. In particular, if φ is a sentence, we simply write $\mathbb{A} \models \varphi$, and say $\mathbb{A}$ is a *model* of φ, or in other words, $\mathbb{A}$ *satisfies* φ.

Let τ be a set of predicates and let φ be any first-order sentence. We say φ is *positive* with respect to τ if it is in the smallest set that contains all atoms, and the negation of atoms not involving any predicate in τ, and that is closed under connectives $\wedge, \vee$ and first-order quantifiers $\exists, \forall$; φ is *negative* with respect to τ if $\neg\varphi$ is equivalent to a sentence positive with respect to τ. By a routine induction on the structure of formulae, we can then obtain the following property:

Proposition 1. *Let τ be a set of predicates and let $\mathbb{A}, \mathbb{B}$ be two structures such that $\mathbb{A} \preccurlyeq_\tau \mathbb{B}$. Let φ be any first-order sentence which is positive (negative) with respect to τ. Then $\mathbb{B}$ is a model of φ if (only if, respectively) $\mathbb{A}$ is a model φ.*

For the propositional case, a slightly stronger version of the above proposition is still true. A *positive clause* is defined to be a finite disjunction of atoms. Given a vocabulary υ, let $\mathrm{PC}(\upsilon)$ be the set of all positive clauses of υ. Now, we give the following property without a proof since it is in fact the same as that of Theorem 1.2.16 in [Chang and Keisler, 1990].

Proposition 2. *Let υ be a set of proposition constants and suppose $\mathbb{M}, \mathbb{N} \subseteq \upsilon$. Then $\mathbb{M} \subseteq \mathbb{N}$ iff for all positive clauses $\gamma \in \mathrm{PC}(\upsilon)$, $\mathbb{N}$ is a model of γ whenever $\mathbb{M}$ is a model of γ.*

2.1 Circumscription

For the notion of circumscription we follow [Lifschitz, 1985], but do not use varying constants. Let φ be a first-order sentence and τ a finite set of predicate constants. For each predicate P in τ we introduce a new predicate variable P^* of the same arity. Let τ^* be the set of all predicate variables P^* where P is in τ. For the sake of convenience, we write $\tau^* = \tau$ for the formula $\wedge_{P\in\tau}\forall\bar{x}(P^*(\bar{x}) \leftrightarrow P(\bar{x}))$, write $\tau^* \leq \tau$ for the formula $\wedge_{P\in\tau}\forall\bar{x}(P^*(\bar{x}) \rightarrow P(\bar{x}))$, and write $\tau^* < \tau$ for the formula $\tau^* \leq \tau \wedge \neg(\tau^* = \tau)$. The *circumscription* $\mathrm{CIRC}(\varphi;\tau)$ of φ with respect to τ is defined to be the second-order sentence $\varphi \wedge \forall\tau^*(\tau^* < \tau \rightarrow \neg\varphi(\tau^*))$, where $\varphi(\tau^*)$ is the formula obtained from φ by substituting the variable P^* for each constant P of τ. A structure $\mathbb{A}$ is called a τ-*minimal model* of φ if $\mathbb{A}$ is a model of φ, and no model $\mathbb{B}$ of φ satisfies $\mathbb{B} \prec_\tau \mathbb{A}$. The following well-known property shows that translation CIRC captures the minimal model semantics.

Proposition 3. *Let φ be a first-order sentence. A structure $\mathbb{A}$ is a τ-minimal model of φ iff it is a model of $\mathrm{CIRC}(\varphi;\tau)$.*

2.2 General Stable Model Semantics

Similarly, the stable model semantics is also defined by a second-order translation. Given a first-order sentence φ and a finite set τ of predicate constants, let $\mathrm{SM}(\varphi;\tau)$ stand for the second-order sentence $\varphi \wedge \forall\tau^*(\tau^* < \tau \rightarrow \neg\mathrm{ST}(\varphi;\tau))$, where the formula $\mathrm{ST}(\varphi;\tau)$ is defined recursively as follows:

- $\mathrm{ST}(P(\bar{t});\tau) = P^*(\bar{t})$ if P is a predicate in τ.
- $\mathrm{ST}(\psi;\tau) = \psi$ if ψ is an atom not in the previous case.
- $\mathrm{ST}(\psi \circ \chi;\tau) = \mathrm{ST}(\psi;\tau) \circ \mathrm{ST}(\chi;\tau)$ if $\circ \in \{\wedge, \vee\}$.
- $\mathrm{ST}(\psi \rightarrow \chi;\tau) = (\psi \rightarrow \chi) \wedge (\mathrm{ST}(\psi;\tau) \rightarrow \mathrm{ST}(\chi;\tau))$.
- $\mathrm{ST}(Qx\psi;\tau) = Qx\mathrm{ST}(\psi;\tau)$ if $Q \in \{\forall, \exists\}$.

A structure $\mathbb{A}$ is called a τ-*stable model* of φ if it is a model of $\mathrm{SM}(\varphi;\tau)$. A predicate constant is said to be *intensional* if it occurs in τ. Otherwise, it is *extensional*. For more information about this, please refer to [Ferraris *et al.*, 2011].

2.3 Logic Programs

Every *disjunctive logic program* is a set of *rules* of the form

$$\zeta_1 \wedge \cdots \wedge \zeta_m \rightarrow \zeta_{m+1} \vee \cdots \vee \zeta_n$$

where $0 \leq m \leq n$, and for each integer $m < i \leq n$, ζ_i is an atom without equality; for each integer $1 \leq j \leq m$, ζ_j is a *literal*, i.e., an atom or its negation. The disjunctive part of the rule is called its *head*, and the conjunctive part called its *body*. Let Π be a disjunctive logic program. Then each *intensional predicate* of Π is a predicate constant that occurs in the head of some rule in Π. We say Π is *normal* if the head of each rule contains at most one literal, Π is *plain* if no negative literal involving an intensional predicate occurs in any of its rules, Π is *propositional* if no predicate of positive arity occurs in any of its rules, and Π is *finite* if it contains only a finite set of rules. In particular, unless mentioned otherwise, a logic program is always assumed to be finite.

For convenience, we regard each disjunctive logic program as the conjunction of all sentences $\forall\bar{x}\gamma$ such that γ is a rule in it and that $\bar{x}$ consists of all individual variables occurring in γ. Let Π be a finite disjunctive logic program Π and let τ be the set of all intensional predicates of Π, we let $\mathrm{CIRC}(\Pi)$ and $\mathrm{SM}(\Pi)$ denote the formulae $\mathrm{CIRC}(\Pi;\tau)$ and $\mathrm{SM}(\Pi;\tau)$ respectively. Every τ-minimal (τ-stable) model of Π is simply called a *minimal* (*stable*, respectively) *model* of Π. It is easy to see that $\mathrm{SM}(\Pi)$ is equivalent to $\mathrm{CIRC}(\Pi)$ if Π is plain.

Let $\mathbb{A}$ be a structure and τ a set of predicates. For every k-ary predicate P in τ and every k-tuple $\bar{a}$ on A, we introduce

$P_{\bar{a}}$ as a new proposition constant. Define $\textsc{Ins}(\mathbb{A},\tau)$ to be the set of proposition constants $P_{\bar{a}}$ such that $P(\bar{a})$ is true in $\mathbb{A}$ and P is in τ. Given a rule γ and an assignment α in $\mathbb{A}$, let $\gamma[\alpha]$ be the rule obtained from γ by substituting $P_{\bar{a}}$ for all atoms $P(\bar{t})$ such that $\bar{a}=\alpha(\bar{t})$, let γ^-_{B} be the set of all literals in the body in which no intensional predicate positively occurs, and let γ^+ be the rule obtained from γ by removing all literals in γ^-_{B}. Given a disjunctive logic program Π, let $\Pi^{\mathbb{A}}$ be the set of rules $\gamma^+[\alpha]$ for all assignments α in $\mathbb{A}$ and all rules γ in Π such that α satisfies γ^-_{B} in $\mathbb{A}$. The following proposition asserts that the general stable model semantics for logic programs can be redefined by the first-order GL-reduction defined above.

Proposition 4. *Let Π be a disjunctive logic program with a set τ of intensional predicates. Then an $\upsilon(\Pi)$-structure $\mathbb{A}$ is a stable model of Π iff $\textsc{Ins}(\mathbb{A},\tau)$ is a minimal model of $\Pi^{\mathbb{A}}$.*

Proof. Similar to Theorem 1 in [Ferraris *et al.*, 2011]. $\square$

3 First-Order Expressibility

As mentioned in the first section, there are three kinds of first-order expressibility having been studied. In this section, we study the relationship among them over arbitrary structures.

Definition 1. Given a second-order formula φ, we say φ is *first-order expressible* if it is equivalent to a first-order formula, φ is *pseudo-first-order expressible* if it is equivalent to an existential second-order formula, and φ is *weak-first-order expressible* if it is equivalent to a first-order theory.

Intuitively, the last two kinds of expressibility are strictly stronger than the first one. But surprisingly, the following result shows this is not true for a large family of semantics.

Proposition 5. *Let φ be a universal second-order formula. Then the following statements are equivalent:*

1. *φ is first-order expressible;*
2. *φ is weak-first-order expressible;*
3. *φ is pseudo-first-order expressible.*

Proof. Clearly, both "1$\Longrightarrow$2" and "1$\Longrightarrow$3" are trivial, and "3$\Longrightarrow$1" follows from the fact that every universal second-order formula whose negation is equivalent to a universal second-order formula should be first-order expressible. This fact is an immediate consequence of Craig's interpolation theorem for first-order logic. Please refer to Subsection 2.5.2 of [van Benthem and Doets, 1983] for a statement. Now, it remains to show "2$\Longrightarrow$1". Assume there is a first-order theory Σ that is equivalent to φ, and suppose φ is of the form $\forall\tau\vartheta$ where ϑ is quantifier-free and τ a finite set of predicates. From $\Sigma\models\varphi$ we have $\Sigma\models\vartheta$. By the compactness, there exists a finite subset Σ_0 of Σ such that $\Sigma_0\models\vartheta$, which implies $\Sigma_0\models\varphi$. On the other hand, by the assumption, it is clearly true that $\varphi\models\Sigma_0$. Let ψ be the conjunction of all formulae in Σ_0. Then ψ is a first-order formula equivalent to φ. $\square$

Remark 1. By the above proposition, the three kinds of first-order expressibility then coincide for both circumscription and the general stable model semantics (so also for disjunctive logic programming) since their syntax-translation definitions are universal second-order. In the rest of this paper, we will only consider the standard first-order expressibility.

4 Progression and Boundedness

Firstly, we generalize the fixed point semantics in [Lobo *et al.*, 1992] to non-Herbrand disjunctive logic programs with default negation, and show the new semantics coinciding with the general stable model semantics.

In order to simplify the presentation, each clause and clauses obtained from it by applying the laws of commutation, association and identity for $\vee$ are regarded as the same.

Definition 2. Given a propositional, possibly infinite and plain disjunctive logic program Π, and given a subset Σ of $\textsc{pc}(\upsilon(\Pi))$, let $\Gamma_\Pi(\Sigma)$ be the set of all clauses $H\vee C_1\vee\cdots\vee C_k$ such that $k\geq 0$ and there are a rule $p_1\wedge\cdots\wedge p_k\to H$ in Π and a sequence of positive clauses $C_1\vee p_1,\ldots,C_k\vee p_k$ in Σ.

From the definition, Γ_Π is clearly a function on the subsets of $\textsc{pc}(\upsilon(\Pi))$. Now by the first-order GL-reduction, a progressional operator for first-order logic programs is then defined.

Definition 3. Let Π be a disjunctive logic program and let $\mathbb{A}$ be a structure of $\upsilon(\Pi)$. We define $\Gamma^{\mathbb{A}}_\Pi$ as the function $\Gamma_{\Pi^{\mathbb{A}}}$.

Assume Γ is a unary function on a set. For convenience, we let $\Gamma\uparrow 0$ denote the empty set $\emptyset$, and let $\Gamma\uparrow n$ be the union of $\Gamma\uparrow n-1$ and $\Gamma(\Gamma\uparrow n-1)$ for all integers $n>0$. Moreover, we define $\Gamma\uparrow\omega$ as the union of $\Gamma\uparrow n$ for all integers $n\geq 0$.

Example 1. Let Π be the logic program consisting of rules

$$P(x)\to P(y)\vee P(z)\quad\text{and}\quad P(c),$$

and let $\mathbb{A}$ be a $\{c\}$-structure. Then $\Gamma^{\mathbb{A}}_\Pi\uparrow 1=\{P_a\}$ where $a=c^{\mathbb{A}}$, and for $1<k\leq|A|$, $\Gamma^{\mathbb{A}}_\Pi\uparrow k$ is the set of all clauses of the form $P_{a_1}\vee\cdots\vee P_{a_k}$ such that $a_1,\ldots,a_k\in A$. $\square$

Given a propositional theory Σ, let $\Lambda(\Sigma)$ denote the set of clauses $\{\gamma\in\textsc{pc}(\upsilon(\Sigma)):\Sigma\models\gamma\}$. The following proposition can be regarded as a simulation of Theorem 3.14 in [Lobo *et al.*, 1992] which asserts the soundness and completeness of their fixed point semantics over Herbrand structures.

Proposition 6. *Let Π be a propositional, possibly infinite and plain disjunctive logic program. Then $\Gamma_\Pi\uparrow\omega\equiv\Lambda(\Pi)$.*

This proposition can be proved in a way similar to Theorem 3.14 in [Lobo *et al.*, 1992]. The only thing should be careful is: logic programs in this paper may involve an uncountable number of proposition constants. But it is not a real obstacle since many properties of the propositional logic, including the completeness and compactness, still hold for the uncountable language (see Section 1.2 of [Chang and Keisler, 1990]).

The above proposition implies that our semantics is indeed a generalization of the fixed point semantics in [Lobo *et al.*, 1992]. To achieve our goal, however, more results are needed.

Proposition 7. *Let Π be a propositional, possibly infinite and plain disjunctive logic program without extensional propositions. Then Π and $\Lambda(\Pi)$ have the same set of minimal models.*

Proof. "Subset": Suppose $\mathbb{M}$ is a minimal model of Π. As $\mathbb{M}$ is clearly a model of Π, by the definition of Λ, $\mathbb{M}$ should satisfy each clause in $\Lambda(\Pi)$. To obtain a contradiction, we assume $\mathbb{M}$ is not a minimal model of $\Lambda(\Pi)$. This means that there must be a model $\mathbb{N}$ of $\Lambda(\Pi)$ such that $\mathbb{N}\subsetneq\mathbb{M}$. Let Σ be the set $\{\neg C: C\in\textsc{pc}(\upsilon(\Pi))\wedge\mathbb{N}\models\neg C\}$. We claim that $\Pi\cup\Sigma$ is satisfiable. Let us first assume the claim is true. Let

$\mathbb{N}'$ be a model of $\Pi \cup \Sigma$. For every clause $C \in \textsc{pc}(\upsilon(\Pi))$, it is clear that, if $\mathbb{N}$ satisfies $\neg C$, then $\neg C \in \Sigma$, which implies $\mathbb{N}'$ satisfies $\neg C$ too. By Proposition 2, it implies $\mathbb{N}' \subseteq \mathbb{N} \subsetneq \mathbb{M}$. But this is impossible since $\mathbb{M}$ is a minimal model of Π.

Now we prove the claim. To obtain a contradiction, assume that $\Pi \cup \Sigma$ is unsatisfiable. By the compactness, there is then a finite subset Σ_0 of Σ such that $\Pi \cup \Sigma_0$ is unsatisfiable. Suppose $\Sigma_0 = \{\neg C_1, \dots, \neg C_k\}$ for some integer k. It must hold that $\Pi \models C_1 \vee \cdots \vee C_k$. Let $C_0 = C_1 \vee \cdots \vee C_k$. It is obvious that $C_0 \in \textsc{pc}(\upsilon(\Pi))$. By the definition of Λ and the previous conclusion, we then have that $C_0 \in \Lambda(\Pi)$. Since $\mathbb{N}$ is a model of $\Lambda(\Pi)$, C_0 should be satisfied by $\mathbb{N}$. Consequently, there should be an integer $1 \le i \le k$ such that $\mathbb{N}$ satisfies C_i, which leads to a contradiction since $C_i \in \Sigma$.

"Superset": Suppose $\mathbb{M}$ is a minimal model of $\Lambda(\Pi)$. Let $\mathbb{N}$ be any proper subset of $\mathbb{M}$. By the assumption, $\mathbb{N}$ does not satisfy $\Lambda(\Pi)$. By the definition of Λ, we can then conclude that $\mathbb{N}$ does not satisfy Π too. Therefore, it remains to show that $\mathbb{M}$ is a model of Π. Let $\Sigma = \{\neg C : C \in \textsc{pc}(\upsilon(\Pi)) \wedge \mathbb{M} \models \neg C\}$. By a similar argument using in the proof of the above claim, we can show that $\Pi \cup \Sigma$ is satisfiable. Let $\mathbb{N}'$ be a model of $\Pi \cup \Sigma$. For every clause $C \in \textsc{pc}(\upsilon(\Pi))$, if $\mathbb{M} \models \neg C$, we have $C \in \Sigma$ by the definition, which implies that $\mathbb{N}'$ satisfies $\neg C$. By Proposition 2, we can then conclude $\mathbb{N}' \subseteq \mathbb{M}$. As we just proved, any proper subset of $\mathbb{M}$ is not a model of Π. So, it must be true that $\mathbb{N}' = \mathbb{M}$, which implies that $\mathbb{M}$ is a model of Π. This then completes the proof. $\square$

Now we show that the stable model semantics for disjunctive logic programs has a progression characterization.

Theorem 1. *Let Π be a disjunctive logic program and let τ be the set of all intensional predicates of Π. Let $\mathbb{A}$ be an arbitrary structure of $\upsilon(\Pi)$. Then $\mathbb{A}$ is a stable model of Π if and only if $\textsc{Ins}(\mathbb{A}, \tau)$ is a minimal model of $\Gamma_\Pi^\mathbb{A} \uparrow \omega$.*

Proof. Simply by the equivalence of following statements:

1. $\mathbb{A}$ is a stable model of Π;
2. $\textsc{Ins}(\mathbb{A}, \tau)$ is a minimal model of $\Pi^\mathbb{A}$;
3. $\textsc{Ins}(\mathbb{A}, \tau)$ is a minimal model of $\Lambda(\Pi^\mathbb{A})$;
4. $\textsc{Ins}(\mathbb{A}, \tau)$ is a minimal model of $\Gamma_{\Pi^\mathbb{A}} \uparrow \omega$;
5. $\textsc{Ins}(\mathbb{A}, \tau)$ is a minimal model of $\Gamma_\Pi^\mathbb{A} \uparrow \omega$.

Herein, the equivalence of statements 1 and 2 is immediate by Proposition 4; 2 and 3 by Proposition 7; 3 and 4 by Proposition 6; and 4 and 5 by the definition. $\square$

Next, we show that the progression can be defined in a first-order way. This will play an important role in the proof of our main result. Some definitions and notations are given firstly.

Every individual variable that will be used in logic programs is supposed to be among $v_0, v_1, v_2, \dots$. For each variable v_i, i is called its *index*. Let V be any set of individual variables. A *renaming function* of V is an injective mapping that maps each variable in V to an individual variable. Given a renaming function λ of V and an expression (including a term or a formula) ϖ, let $\lambda(\varpi)$ be the expression obtained from ϖ by substituting $\lambda(x)$ for all variables x in V.

In many cases, what variables are used in a logic program is not important, and different renaming functions may play the same role. Therefore, a method is then needed to make choices among such kinds of renaming functions. A natural way to do this is by arming with a linear order on renaming functions and then choosing the least one satisfying a certain condition. We define a linear order on renaming functions as follows. Let λ_1 and λ_2 be two renaming function of V. Given $k = 1$ or 2, let S_k be the sequence of all indices of $\lambda_k(x)$ for all $x \in V$ which is ordered by the index of x, i.e., if $i < j$ and $v_i, v_j \in V$, then the position of index of $\lambda_k(v_i)$ in S_k should be in front of that of $\lambda_k(v_j)$. We say λ_1 is *less than* λ_2 iff the sequence S_1 is less than S_2 in the lexicographic order.

Definition 4. Let Π be a disjunctive logic program. We let $\Delta^0(\Pi) = \emptyset$, and for each integer $n > 0$, let $\Delta^n(\Pi)$ be the union of $\Delta^{n-1}(\Pi)$ and the set of all rules

$$\eta \wedge \eta_1^\diamond \wedge \cdots \wedge \eta_k^\diamond \rightarrow \vartheta \vee \lambda_1(\vartheta_1) \vee \cdots \vee \lambda_k(\vartheta_k)$$

that satisfy all the following conditions:

1. there is a rule $\gamma = P_1(\bar{t}_1) \wedge \cdots \wedge P_k(\bar{t}_k) \wedge \eta \rightarrow \vartheta$ in Π such that no intensional predicate positively occurs in η,
2. for each integer $1 \le i \le k$, there is a rule $\gamma_i = \eta_i \rightarrow \vartheta_i \vee P_i(\bar{s}_i)$ in $\Delta^{n-1}(\Pi)$, and
3. for each integer $1 \le i \le k$, λ_i is the least renaming function of V such that no individual variable occurring in the formulae $\lambda_1(\gamma_1), \dots, \lambda_i(\gamma_i)$ occurs in γ, and $\eta_i^\diamond$ is the formula $\lambda_i(\eta_i) \wedge (\lambda_i(\bar{s}_i) = \bar{t}_i)$, where V is the set of all individual variables occurring in $\gamma, \gamma_1, \dots, \gamma_k$.

Moreover, let $\Delta(\Pi)$ denote the union of $\Delta^n(\Pi)$ for all $n \ge 0$.

Example 2. (Example 1 continued) Let Π be the same as Example 1 and suppose x, y, z are v_0, v_1, v_2 respectively. Then

$$\begin{array}{rcl}
\Delta^1(\Pi) & = & \{P(c)\}\,, \\
\Delta^2(\Pi) & = & \{P(c), \quad c = v_0 \rightarrow P(v_1) \vee P(v_2)\}\,, \\
\Delta^3(\Pi) & = & \left\{ \begin{array}{r} P(c), \quad c = v_0 \rightarrow P(v_1) \vee P(v_2), \\ c = v_3 \wedge v_5 = v_0 \rightarrow P(v_1) \vee P(v_2) \vee P(v_4), \\ c = v_3 \wedge v_4 = v_0 \rightarrow P(v_1) \vee P(v_2) \vee P(v_5) \end{array} \right\}
\end{array}$$

and so on. Clearly, $\Delta^n(\Pi)$ is a finite set for each $n \ge 0$. $\square$

Intuitively, $\Delta^n(\Pi)$ is the set of formulae derivable in n steps by some rules. The following property is not surprise.

Proposition 8. *Let Π be a disjunctive logic program. Let $\mathbb{A}$ be an $\upsilon(\Pi)$-structure. Then $\Gamma_\Pi^\mathbb{A} \uparrow n = \Delta^n(\Pi)^\mathbb{A}$ for all $n \ge 0$.*

The proof of this proposition is slightly tedious, but its intuitive meaning is very clear. Due to the limit of space, we omit the proof here. A complete proof through an induction on n will be available in the full version of this paper.

Now, we define the boundedness of a logic program.

Definition 5. Let Π be a plain disjunctive logic program and τ the set of all intensional predicates of Π. Let $\mathcal{C}$ be a class of τ-structures. Given an integer $k \ge 0$, Π is *k-bounded* on $\mathcal{C}$ if for every structure $\mathbb{A}$ in $\mathcal{C}$, $\Gamma_\Pi^\mathbb{A} \uparrow \omega$ is equivalent to $\Gamma_\Pi^\mathbb{A} \uparrow k$. Π is *bounded* on $\mathcal{C}$ if it is k-bounded on $\mathcal{C}$ for some integer $k \ge 0$. If $\mathcal{C}$ is the class of all τ-structures and Π is bounded (k-bounded) on $\mathcal{C}$, we simply say Π is *bounded* (*k-bounded*).

In this definition, please notice that $\Gamma_\Pi^\mathbb{A} \uparrow \omega$ and $\Gamma_\Pi^\mathbb{A} \uparrow n$ are required to be equivalent. It seems more natural to define boundedness by the set-identity relation. But the following example shows that this is not enough to capture the first-order expressibility of plain disjunctive logic programs.

Example 3. (Example 1 continued) Let Π be the program in Example 1. It is clear that $\textsc{Circ}(\Pi)$ is equivalent to the first-order sentence $\forall x P(x)$. Let $\mathbb{A}$ be any $\{c\}$-structure. Then $\Gamma_{\Pi}^{\mathbb{A}} \uparrow n$ is a proper subset of $\Gamma_{\Pi}^{\mathbb{A}} \uparrow \omega$ for each integer $n < |A|$. On the other hand, $\Gamma_{\Pi}^{\mathbb{A}} \uparrow 2$ is equivalent to $\Gamma_{\Pi}^{\mathbb{A}} \uparrow \omega$ since the clause $P_a \in \Gamma_{\Pi}^{\mathbb{A}} \uparrow 2$ for each a in A. Hence, Π is bounded. □

Definition 6. Let Π be a plain disjunctive logic program. Given an integer $k \geq 0$, we say Π is *k-finitary* with respect to derivation if $\Delta(\Pi)$ is equivalent to $\Delta^k(\Pi)$. We say Π is *finitary* with respect to derivation if it is k-finitary with respect to derivation for some integer $k \geq 0$.

The following theorem follows from Proposition 8.

Theorem 2. *A plain disjunctive logic program is k-bounded if and only if it is k-finitary with respect to derivation.*

Now, let us consider non-plain disjunctive logic programs. Let Π be such a program and let τ be the set of all intensional predicates of Π. Lin's transformation translates Π to a plain disjunctive logic program Π^* which is obtained from Π by, for each P of τ, substituting P^* for all positive occurrences of P in the head or the body of rules [Lin, 1991]. By the second-order translation, it is easy to see that $\mathrm{SM}(\Pi)$ is equivalent to $\exists \tau^*(\textsc{Circ}(\Pi^*) \wedge \tau^* = \tau)$. So, a natural definition of boundedness for non-plain programs can be defined by this transformation. Unfortunately, the following example shows that it does not capture the first-order expressibility.

Example 4. Let Π be the following logic program:

$$\begin{array}{rclrcl} E(x,y) & \to & R(x,y), & E(x,y) & \to & P(x,y), \\ E(x,y) \wedge R(y,z) & \to & R(x,z), & \neg P(x,y) & \to & P(x,y). \end{array}$$

It is easy to verify that $\mathrm{SM}(\Pi)$ is equivalent to the first-order sentence $\forall xy(E(x,y) \wedge P(x,y) \wedge R(x,y))$. Let $n > 0$ be an integer, and let $\mathbb{A}_n$ be a structure encoding the linear order of length n. Then n is the least integer k such that $\Gamma_{\Pi^*}^{\mathbb{A}_n} \uparrow k$ is equivalent to $\Gamma_{\Pi^*}^{\mathbb{A}_n} \uparrow \omega$, which implies Π^* is unbounded. □

Therefore, we have to choose a weaker definition for the boundedness of non-plain disjunctive logic programs. By the next proposition, it can assure the first-order expressibility.

Definition 7. Let Π be a non-plain disjunctive logic program and let τ be the set of all intensional predicates of Π. Let $\mathcal{S}(\Pi)$ be the union of the class of stable models of Π and classes of stable models of $\Delta^k(\Pi)$ for all integers $k \geq 1$. We then say Π is *bounded* iff Π^* is bounded on $\mathcal{S}(\Pi)$.

Proposition 9. *Let Π be a bounded and plain (non-plain) disjunctive logic program. Then Π and $\Delta^k(\Pi)$ have the same class of minimal (stable, respectively) models for some $k \geq 0$.*

Proof. We only consider the case that Π is a non-plain disjunctive logic program. For the plain case, a similar proof can be easily obtained. By the definition, there must be some integer $k \geq 0$ such that Π^* is k-bounded over $\mathcal{S}(\Pi)$. Now, it suffices to prove the equivalence of following statements:

1. $\mathbb{A}$ is a stable model of Π;
2. $\textsc{Ins}(\mathbb{A}, \tau)$ is a minimal model of $\Gamma_{\Pi}^{\mathbb{A}} \uparrow \omega$;
3. $\textsc{Ins}(\mathbb{A}, \tau)$ is a minimal model of $\Gamma_{\Pi}^{\mathbb{A}} \uparrow k$;
4. $\textsc{Ins}(\mathbb{A}, \tau)$ is a minimal model of $\Delta^k(\Pi)^{\mathbb{A}}$;
5. $\mathbb{A}$ is a stable model of $\Delta^k(\Pi)$.

Herein, the equivalence of statements 1 and 2 follows from Theorem 1; 2 and 3 from the assumption that Π is k-bounded; 3 and 4 from Proposition 8; 4 and 5 from Proposition 4. □

5 A Preservation Theorem

Let Π be a disjunctive logic program. The *dependency graph* of Π is the directed graph whose vertices are predicates in $\upsilon(\Pi)$ and that consists of all edges from P to Q such that P and Q positively occur in the head and the body of a rule in Π respectively. We say Π is *recursion-free* if every intensional predicate has no positive occurrence in the body of any rule in Π, and say Π is *loop-free* if its dependency graph is acyclic.

Now we give the statement of our main theorem. The rest of this section is then devoted to prove the main theorem.

Main Theorem. *Let Π be a disjunctive logic program. If Π is plain, the following statements are then equivalent:*

1. *Π is bounded;*
2. *Π is finitary with respect to derivation;*
3. *There is a recursion-free disjunctive logic program Π_0 such that $\mathrm{SM}(\Pi) \equiv \mathrm{SM}(\Pi_0)$;*
4. *There is a loop-free disjunctive logic program Π_0 such that $\mathrm{SM}(\Pi) \equiv \mathrm{SM}(\Pi_0)$;*
5. *$\mathrm{SM}(\Pi)$ is first-order expressible;*
6. *$\mathrm{SM}(\Pi)$ is weak-first-order expressible;*
7. *$\mathrm{SM}(\Pi)$ is pseudo-first-order expressible.*

Otherwise, the statements 5–7 are equivalent, and the statement 1 implies all statements among 3–7.

Before proving the theorem, we introduce a tool from classical model theory - the ultraproduct. Let I be a nonempty set and D a set of subsets of I. Then D is a *filter* over I if:

1. if both X and Y are in D then $X \cap Y$ is in D, and
2. if X is in D and $X \subseteq Y \subseteq I$ then Y is also in D.

Moreover, D is an *ultrafilter* over I if D is a filter over I and for every subset X of I, X is in D iff $I \setminus X$ is not in D.

Assume D is an ultrafilter over I. Let υ be a vocabulary, and let $\mathbb{A}_i (i \in I)$ be a family of υ-structures. Let $\prod_{i \in I} A_i$ be the set of all functions π with domain I such that $\pi(i) \in A_i$ for all $i \in I$. For all $\pi_1, \pi_2 \in \prod_{i \in I} A_i$, we write $\pi_1 \sim_D \pi_2$ if the set $\{i \in I : \pi_1(i) = \pi_2(i)\}$ is in D. It is clear that $\sim_D$ is an equivalence relation. If $\pi \in \prod_{i \in I} A_i$ then let π^D be the equivalence class π with respect to $\sim_D$. Let $\prod_D A_i$ denote the set of equivalence classes π^D for all $\pi \in \prod_{i \in I} A_i$. The *ultraproduct of $\mathbb{A}_i (i \in I)$ module D*, written as $\prod_D \mathbb{A}_i$, is defined to be the υ-structure satisfying the following conditions:

1. the domain of $\prod_D \mathbb{A}_i$ is $\prod_D A_i$;
2. for each k-ary predicate P in υ, $P^{\prod_D \mathbb{A}_i}$ consists of all the tuples $(\pi_1^D, \ldots, \pi_k^D)$ such that $\pi_1^D, \ldots, \pi_k^D \in \prod_D A_i$ and $\{i \in I : (\pi_1(i), \ldots, \pi_k(i)) \in P^{\mathbb{A}_i}\} \in D$;
3. for each k-ary function f in υ and for all $\pi_1^D, \ldots, \pi_k^D$ in $\prod_D A_i$, $f^{\prod_D \mathbb{A}_i}(\pi_1^D, \ldots, \pi_k^D) = \pi^D$ where π is defined by $\pi(i) = f^{\mathbb{A}_i}(\pi_1(i), \ldots, \pi_k(i))$ for all $i \in I$.

Let $\mathcal{C}$ be an arbitrary class of structures. We say $\mathcal{C}$ is *closed* under ultraproducts if $\prod_D \mathbb{A}_i$ is in $\mathcal{C}$ whenever D is an ultrafilter over some nonempty set I and $\mathbb{A}_i$ is in $\mathcal{C}$ for every index i in I. We say $\mathcal{C}$ is *definable* by a first-order sentence φ (a first-order theory Σ) if $\mathcal{C}$ is exactly the class of all models of φ (of Σ). The following proposition immediately follows from Theorem 4.1.12 in [Chang and Keisler, 1990].

Proposition 10. *A class of structures is definable by a first-order sentence if and only if it is definable by a first-order theory and its complement is closed under ultraproducts.*

The following property, which says that every consistent universal first-order theory has a minimal model, is an immediate corollary of Property 1.3.2 in [Bossu and Siegel, 1985].

Proposition 11. *Let Σ be a possibly infinite set of universal first-order sentences, and let $\mathbb{A}$ be a model of Σ. Then there is a minimal model $\mathbb{B}$ of Σ such that $\mathbb{B} \preccurlyeq_\tau \mathbb{A}$.*

For the notation $\Delta(\cdot)$ defined in the previous section, the following property immediately follows from the definition.

Lemma 1. *Let Π be a plain disjunctive logic program. Then for every rule γ in $\Delta(\Pi)$, Π entails γ in classical logic.*

With these properties, we can then prove the following result, which plays a critical role in the proof of Main Theorem.

Proposition 12. *Let Π be a plain disjunctive logic program. If $\textsc{Circ}(\Pi)$ is equivalent to a first-order sentence, then Π is finitary with respect to derivation.*

Proof. We first prove the statement that there is a finite subset of $\Delta(\Pi)$ which is equivalent to a first-order sentence. To obtain a contradiction, we assume it is not true. By Proposition 10, there should be an index set I, an ultrafilter D over I and a family of structures $\mathbb{A}_i (i \in I)$ of $\upsilon(\Pi)$ such that $\mathbb{A}_i$ does not satisfy $\Delta(\Pi)$ for any index $i \in I$, but $\prod_D \mathbb{A}_i$ satisfies $\Delta(\Pi)$. Let τ be the set of all intensional predicates of Π. By Proposition 11, there must be a τ-minimal model $\mathbb{B}$ of $\Delta(\Pi)$ such that $\mathbb{B} \preccurlyeq_\tau \prod_D \mathbb{A}_i$ $(*)$. For each index $i \in I$, let $\mathbb{B}_i$ be a structure obtained from $\mathbb{A}_i$ by removing tuples $(\pi_1(i), \dots, \pi_k(i))$ from $P^{\mathbb{A}_i}$ if $k \geq 0$, P is a k-ary predicate in τ and the tuple $(\pi_1^D, \dots, \pi_k^D)$ is in $P^{\prod_D \mathbb{A}_i}$ but not in $P^{\mathbb{B}}$. Then it is clear that, for all indices $i \in I$, we have $\mathbb{B}_i \preccurlyeq_\tau \mathbb{A}_i$. By Proposition 1, we conclude that, for each $i \in I$, $\mathbb{B}_i$ does not satisfy $\Delta(\Pi)$. According to Lemma 1, this implies that $\mathbb{B}_i$ does not satisfy Π. Consequently, $\mathbb{B}$ cannot be any model of $\textsc{Circ}(\Pi)$. According to the assumption, $\textsc{Circ}(\Pi)$ is equivalent to a first-order sentence. So, by Proposition 10 again, $\prod_D \mathbb{B}_i$ is not a model of $\textsc{Circ}(\Pi)$ too. On the other hand, by the definition of ultraproduct, it is easy to check that $\prod_D \mathbb{B}_i = \mathbb{B}$, which contradicts with the conclusion $(*)$.

With this statement, there is then a first-order sentence φ that is equivalent to $\Delta(\Pi)$. And by the compactness, there must be a finite subset Φ of $\Delta(\Pi)$ such that $\Phi \models \phi$, which immediately implies that $\Phi \equiv \phi$. Let n be the smallest integer such that $\Phi \subseteq \Delta^n(\Pi)$, then $\Delta^n(\Pi)$ should be equivalent to $\Delta(\Pi)$. Hence, Π is finitary with respect to derivation. $\square$

Now we are in the position to prove Main Theorem.

Proof of Main Theorem. The equivalence of statements 5–7 immediately follows from Proposition 5 and Remark 1. The direction of "3$\Longrightarrow$4" is trivially true, "1$\Longrightarrow$3" follows from Proposition 9, "4$\Longrightarrow$5" follows from Theorem 11 of [Ferraris *et al.*, 2011] and the fact that every loop-free program is tight. In particular, for the case that Π is plain, the equivalence of statements 1 and 2 is immediately obtained by Theorem 2, and the direction of "5$\Longrightarrow$2" is by Proposition 12. $\square$

Remark 2. For the non-plain programs, Main Theorem shows that the boundedness is a sufficient condition for the first-order expressibility. However, it is not clear whether or not it is also a necessary condition. The main difficulty of generalizing our preservation theorem to non-plain case is: it seems hard to find a replacement for Proposition 11. Therefore, it is a challenge to find a boundedness-like characterization for the first-order expressibility of non-plain logic programs.

Remark 3. Over finite structures, the boundedness of plain logic programs is not enough to capture the standard first-order expressibility. A counterexample, which does not involve any default negations and disjunctions, could be found in the proof of Theorem 11.1 in [Ajtai and Gurevich, 1994].

6 Related Works and Conclusion

The boundedness characterization of the first-order expressibility for function-free definite logic programs was independently discovered by [Kolaitis and Papadimitriou, 1990] and [Ajtai and Gurevich, 1994]. Our preservation theorem extends theirs in two directions: allowing disjunctions in the head of rules, and allowing functions in terms. It seems that their approaches cannot be applied to either extension.

Minker *et al.* developed a fixed point semantics for plain disjunctive logic programs over Herbrand structures [1990; 1992]; Zhang and Zhou proposed a progression semantics for normal logic programs [2010], and proved their boundedness defined on this semantics is a necessary condition for the first-order expressibility. Both works are limited to certain classes of logic programs and even a certain class of structures. Our semantics is in fact a natural generalization of both.

There are two kinds of progression semantics which were proposed for disjunctive logic programs with default negation [Leone *et al.*, 1997; Zhou and Zhang, 2011]. The first is based on a technique of unfounded sets and works on Herbrand structures, and the second progresses disjunctive logic programs in a parallel way. Both semantics are significantly different from ours. It is not clear whether or not their semantics can be used to capture the first-order expressibility.

The preservation theorem presented in this paper is an interesting model-theoretic property for disjunctive logic programs, and gives us a precise picture on the relationship between disjunctive logic programs and classical logic. It also provides us a manageable condition for identifying first-order expressible fragments. Moreover, the progression semantics and boundedness proposed here may shed light on the static analysis and optimization of disjunctive logic programs.

Acknowledgement

This publication was made possible by the support of an NPRP grant (NPRP 09-079-1-013) from the Qatar National Research Fund (QNRF). The statements made herein are solely the responsibility of the authors.

References

[Abiteboul *et al.*, 1995] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley Publishing Company, New York, 1995.

[Ajtai and Gurevich, 1994] M. Ajtai and Y. Gurevich. Datalog vs first-order logic. *Journal of Computer and System Sciences*, 49:562–588, 1994.

[Asuncion *et al.*, 2012a] V. Asuncion, F. Lin, Y. Zhang, and Y. Zhou. Ordered completion for first-order logic programs on finite structures. *Artificial Intelligence*, 177–179:1–24, 2012.

[Asuncion *et al.*, 2012b] V. Asuncion, Y. Zhang, and Y. Zhou. Ordered completion for logic programs with aggregates. In *Proc. AAAI*, pages 691–697, 2012.

[Baral, 2003] C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, Cambridge, 2003.

[Bartholomew and Lee, 2010] M. Bartholomew and J. Lee. A decidable class of groundable formulas in the general theory of stable models. In *Proc. KR*, 2010.

[Barwise and Moschovakis, 1978] J. Barwise and Y. Moschovakis. Global inductive definability. *The Journal of Symbolic Logic*, 43(3):521–534, 1978.

[Bossu and Siegel, 1985] G. Bossu and P. Siegel. Saturation, nonmonotonic reasoning and the closed-world assumption. *Artificial Intelligence*, 25(1):13–63, 1985.

[Cabalar *et al.*, 2009] P. Cabalar, D. Pearce, and A. Valverde. A revised concept of safety for general answer set programs. In *Proc. LPNMR*, pages 58–70, 2009.

[Chang and Keisler, 1990] C. C. Chang and H. J. Keisler. *Model Theory*. North Holland, New York, 3 edition, 1990.

[Chen *et al.*, 2006] Y. Chen, F. Lin, Y. Wang, and M. Zhang. First-order loop formulas for normal logic programs. In *Proc. KR*, pages 298–307, 2006.

[Chen *et al.*, 2010] Y. Chen, Y. Zhang, and Y. Zhou. First-order indefinability of answer set programs on finite structures. In *Proc. AAAI*, pages 285–290, 2010.

[Chen *et al.*, 2011] Y. Chen, F. Lin, Y. Zhang, and Y. Zhou. Loop-separable programs and their first-order definability. *Artificial Intelligence*, 175:890–913, 2011.

[Chomicki and Subrahmanian, 1990] J. Chomicki and V. S. Subrahmanian. Generalized closed world assumption is Π^0_2-complete. *Information Processing Letters*, 35:289–291, 1990.

[Eiter and Gottlob, 1997] T. Eiter and G. Gottlob. Expressiveness of stable model semantics for disjunctive logic programs with functions. *The Journal of Logic Programming*, 33:167–178, 1997.

[Eiter *et al.*, 1997] T. Eiter, G. Gottlob, and H. Mannila. Disjunctive datalog. *ACM Transactions on Database Systems*, 22:364–418, 1997.

[Ferraris *et al.*, 2011] P. Ferraris, J. Lee, and V. Lifschitz. Stable models and circumscription. *Artificial Intelligence*, 175:236–263, 2011.

[Gelfond and Lifschitz, 1988] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *Proc. ICLP/SLP*, pages 1070–1080, 1988.

[Kolaitis and Papadimitriou, 1990] P. G. Kolaitis and C. H. Papadimitriou. Some computational aspects of circumscription. *Journal of the ACM*, 37:1–14, 1990.

[Lee and Meng, 2011] J. Lee and Y. Meng. First-order stable model semantics and first-order loop formulas. *Journal of Artificial Intelligence Research*, 42:125–180, 2011.

[Leone *et al.*, 1997] N. Leone, P. Rullo, and F. Scarcello. Disjunctive stable models: Unfounded sets, fixpoint semantics and computation. *Information and Computation*, 135(2):69–112, 1997.

[Lifschitz and Yang, 2012] V. Lifschitz and F. Yang. Lloyd-topor completion and general stable models. In *5th Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP)*, 2012.

[Lifschitz, 1985] V. Lifschitz. Computing circumscription. In *Proc. IJCAI*, pages 121–127, 1985.

[Lin and Zhou, 2011] F. Lin and Y. Zhou. From answer set logic programming to circumscription via logic of GK. *Artificial Intelligence*, 175(1):264–277, 2011.

[Lin, 1991] F. Lin. *A Study of Nonmonotonic Reasoning*. PhD thesis, Stanford University, USA, 1991.

[Lobo *et al.*, 1992] J. Lobo, J. Minker, and A. Rajasekar. *Foundations of Disjunctive Logic Programming*. The MIT Press, Cambridge, 1992.

[Minker and Rajasekar, 1990] J. Minker and A. Rajasekar. A fixpoint semantics for disjunctive logic programs. *Journal of Logic Programming*, 9(1):45–74, 1990.

[Moschovakis, 1974] Y. Moschovakis. *Elementary Induction on Abstract Structures*. North Holland, Amsterdam, 1974.

[van Benthem and Doets, 1983] J. van Benthem and K. Doets. Higher-order logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 1, pages 275–329. Reidel, Dordrecht, 1983.

[Zhang and Ying, 2010] H. Zhang and M. Ying. Decidable fragments of first-order language under stable model semantics and circumscription. In *Proc. AAAI*, 2010.

[Zhang and Zhou, 2010] Y. Zhang and Y. Zhou. On the progression semantics and boundedness of answer set programs. In *Proc. KR*, 2010.

[Zhou and Zhang, 2011] Y. Zhou and Y. Zhang. Progression semantics for disjunctive logic programs. In *Proc. AAAI*, 2011.