Artificial Intelligence 175 (2011) 236–263

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Stable models and circumscription

Paolo Ferraris [a], Joohyung Lee [b], Vladimir Lifschitz [c,*]

[a] Google, Inc., 1600 Amphitheatre Parkway, Mountain View, CA 94043, United States
[b] Department of Computer Science and Engineering, Arizona State University, 699 South Mill Avenue, Tempe, AZ 85281, United States
[c] Department of Computer Sciences, University of Texas at Austin, 1 University Station C0500, Austin, TX 78712, United States

ARTICLE INFO

Article history:
Available online 13 April 2010

Keywords:
Answer set programming
Circumscription
Nonmonotonic reasoning
Program completion
Stable models

ABSTRACT

The concept of a stable model provided a declarative semantics for Prolog programs with negation as failure and became a starting point for the development of answer set programming. In this paper we propose a new definition of that concept, which covers many constructs used in answer set programming and, unlike the original definition, refers neither to grounding nor to fixpoints. It is based on a syntactic transformation similar to parallel circumscription.

1. Introduction

Answer set programming (ASP) is a form of declarative logic programming oriented towards knowledge-intensive search problems, such as product configuration and planning. It was identified as a new programming paradigm ten years ago [25, 29], and it has found by now a number of serious applications. An ASP program consists of rules that are syntactically similar to Prolog rules, but the computational mechanisms used in ASP are different: they use the ideas that have led to the creation of fast satisfiability solvers for propositional logic [11].

ASP is based on the concept of a stable model [9]. According to the definition, to decide which sets of ground atoms are "stable models" of a given set of rules we first replace each of the given rules by all its ground instances. Then we verify a fixpoint condition that is similar to the conditions employed in the semantics of default logic [33] and autoepistemic logic [28] (see [19, Sections 4, 5] for details).

In this paper we investigate a new approach to defining the concept of a stable model. It is based on a syntactic transformation similar to circumscription [26,27]. The new definition refers neither to grounding nor to fixpoints. It turns out to be more general, in a number of ways, than the original definition.

This treatment of stable models may be of interest for several reasons. First, it provides a new perspective on the place of stable models within the field of nonmonotonic reasoning. We can distinguish between "fixpoint" nonmonotonic formalisms, such as default logic and autoepistemic logic, and "translational" formalisms, such as program completion [1] and circumscription. In the past, stable models were seen as part of the "fixpoint tradition." The remarkable similarity between the new definition of a stable model and the definition of circumscription is curious from this point of view.

Second, we expect that the new definition of a stable model will provide a unified framework for useful answer set programming constructs that have been defined and implemented by different research groups. For instance, it may help us combine choice rules in the sense of LPARSE [34] with aggregates in the sense of DLV [3]. A step in this direction is described in [14].

* Corresponding author.
E-mail address: vl@cs.utexas.edu (V. Lifschitz).

0004-3702/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.artint.2010.04.011

Third, our definition is applicable to non-Herbrand models. In such a model, different ground terms may have the same value. This may be useful for knowledge representation purposes; we may wish to write, for instance:

$$Father(Jack) = Father(Jane).$$

This possibility is related also to the use of arithmetic functions in ASP, when different ground terms may have the same value ($2 + 2 = 1 + 3$).

The new definition of a stable model is introduced in Section 2, and its relation to the original definition is discussed in Section 3. Several useful theorems about the new concept are stated in Section 4. Then we extend the idea of strong equivalence to this framework (Section 5), relate general stable models to program completion (Section 6), and define "pointwise stable models," which are similar to pointwise circumscription (Section 7). In Section 8, we show how our theory of stable models handles strong (or classical) negation, and Section 9 discusses related work. Proofs of theorems are collected in Appendix A.

To make the presentation more self-contained, we include brief reviews of parallel and pointwise circumscription (Sections 2.2 and 7.1) and of two approaches to the stable model semantics proposed earlier (Section 3.1).

This article is an extended version of the conference paper [6].

2. Stable models

2.1. Logic programs as first-order sentences

The concept of a stable model will be defined here for first-order sentences,[1] possibly containing function constants and equality. Logic programs are viewed in this paper as alternative notation for first-order sentences of special types. For instance, we treat the logic program

$$\begin{aligned}&p(a,a),\\&p(a,b),\\&q(x) \leftarrow p(x,y)\end{aligned} \tag{1}$$

as shorthand for

$$p(a,a) \wedge p(a,b) \wedge \forall xy\big(p(x,y) \rightarrow q(x)\big). \tag{2}$$

The constraint

$$\leftarrow p(x), \textit{not}\ q(x) \tag{3}$$

is identified with the formula

$$\forall x \neg\big(p(x) \wedge \neg q(x)\big),$$

and the disjunctive rule

$$p(x); q(y) \leftarrow r(x,y)$$

with

$$\forall xy\big(r(x,y) \rightarrow \big(p(x) \vee q(y)\big)\big).$$

As another example, take the choice rule

$$\big\{p(x)\big\} \leftarrow q(x).$$

It says, informally speaking: for every x such that $q(x)$, choose arbitrarily whether or not to include $p(x)$ in the stable model. We can treat this rule as shorthand for

$$\forall x\big(q(x) \rightarrow \big(p(x) \vee \neg p(x)\big)\big). \tag{4}$$

This formula is logically valid, so that appending it as a conjunctive term to any sentence F would not change the class of models of F. But the class of *stable* models of F may change, as we will see, after appending (4).

The next example involves an aggregate. The rule

$$p(x) \leftarrow \#\mathrm{card}\big\{y{:}\ q(x,y)\big\} < 2$$

means intuitively: if the cardinality of the set $\{y{:}\ q(x,y)\}$ is less than 2 then include $p(x)$ in the stable model. We can treat this rule as an abbreviation for the formula

$$\forall x\big(\neg\exists y_1 y_2\big(q(x,y_1) \wedge q(x,y_2) \wedge y_1 \neq y_2\big) \rightarrow p(x)\big). \tag{5}$$

[1] A *sentence* is a formula without free variables.

2.2. Review of circumscription

Since the new definition of a stable model is similar to the definition of parallel circumscription, we will begin with a brief review of the latter.

Both definitions use the following notation. If p and q are predicate constants of the same arity then $p \leqslant q$ stands for the formula

$$\forall \mathbf{x}\big(p(\mathbf{x}) \to q(\mathbf{x})\big),$$

where $\mathbf{x}$ is a tuple of distinct object variables. If $\mathbf{p}$ and $\mathbf{q}$ are tuples $p_1, \dots, p_n$ and $q_1, \dots, q_n$ of predicate constants then $\mathbf{p} \leqslant \mathbf{q}$ stands for the conjunction

$$(p_1 \leqslant q_1) \wedge \dots \wedge (p_n \leqslant q_n),$$

and $\mathbf{p} < \mathbf{q}$ stands for $(\mathbf{p} \leqslant \mathbf{q}) \wedge \neg(\mathbf{q} \leqslant \mathbf{p})$. In second-order logic, we apply the same notation to tuples of predicate variables.

Let $\mathbf{p}$ be a list of distinct predicate constants.[2] The *circumscription operator with the minimized predicates* $\mathbf{p}$, denoted by $\mathrm{CIRC}_{\mathbf{p}}$, is defined as follows: for any first-order formula F, $\mathrm{CIRC}_{\mathbf{p}}[F]$ is the second-order formula

$$F \wedge \neg \exists \mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F(\mathbf{u})\big),$$

where $\mathbf{u}$ is a list of distinct predicate variables of the same length as $\mathbf{p}$, and $F(\mathbf{u})$ is the formula obtained from F by substituting the variables $\mathbf{u}$ for the constants $\mathbf{p}$.[3]

If the list $\mathbf{p}$ is empty then we understand $\mathrm{CIRC}_{\mathbf{p}}[F]$ as F. We will drop the subscript in the symbol $\mathrm{CIRC}_{\mathbf{p}}$ when this does not lead to confusion.

For any sentence F, a $\mathbf{p}$-*minimal* (or simply *minimal*) *model* of F is an interpretation of the underlying signature that satisfies $\mathrm{CIRC}_{\mathbf{p}}[F]$. Since the first conjunctive term of $\mathrm{CIRC}_{\mathbf{p}}[F]$ is F, it is clear that every minimal model of F is a model of F.

Example 1. If F is formula (2) then $\mathrm{CIRC}_{pq}[F]$ is

$$\forall xy\big(p(a,a) \wedge p(a,b) \wedge \big(p(x,y) \to q(x)\big)\big)$$
$$\wedge \neg \exists uv\big(\big((u,v) < (p,q)\big) \wedge \forall xy\big(u(a,a) \wedge u(a,b) \wedge \big(u(x,y) \to v(x)\big)\big)\big).$$

It can be equivalently rewritten without second-order variables as follows:

$$\forall x\big(p(x,y) \leftrightarrow (x=a \wedge y=a) \vee (x=a \wedge y=b)\big) \wedge \forall x\big(q(x) \leftrightarrow x=a\big). \tag{6}$$

Example 2. Let F be the formula

$$\forall xy\big(p(x,y) \to t(x,y)\big) \wedge \forall xyz\big(t(x,y) \wedge t(y,z) \to t(x,z)\big) \tag{7}$$

("p is a subset of t, and t is a transitive relation"). Then $\mathrm{CIRC}_t[F]$ is

$$\forall xy\big(p(x,y) \to t(x,y)\big) \wedge \forall xyz\big(t(x,y) \wedge t(y,z) \to t(x,z)\big)$$
$$\wedge \neg \exists u\big((u < t) \wedge \forall xy\big(p(x,y) \to u(x,y)\big) \wedge \forall xyz\big(u(x,y) \wedge u(y,z) \to u(x,z)\big)\big).$$

This condition cannot be expressed by a first-order formula, but its meaning is straightforward: it says that t is the transitive closure of p.

If we conjoin (7) with

$$p(a,b) \wedge p(b,c) \tag{8}$$

and include both p and t in the list of minimized predicates then the circumscription formula will become expressible in first-order logic as

$$\forall xy\big(p(x,y) \leftrightarrow (x=a \wedge y=b) \vee (x=b \wedge y=c)\big)$$
$$\wedge \forall xy\big(t(x,y) \leftrightarrow (x=a \wedge y=b) \vee (x=b \wedge y=c) \vee (x=a \wedge y=c)\big). \tag{9}$$

[2] In this paper, equality is not considered a predicate constant, so that it is not allowed to be a member of $\mathbf{p}$.

[3] This definition of the circumscription operator allows F to have free variables, unlike the definition from [17]. Similarly, the definition of the stable model operator below is applicable to formulas with free variables, unlike the definition proposed in the conference paper [6].

2.3. *Operator* SM

We will now define the *stable model operator with the intensional predicates* $\mathbf{p}$, denoted by $\mathrm{SM}_{\mathbf{p}}$. Some details of the definition depend on which propositional connectives and quantifiers are treated as primitives, and which of them are viewed as abbreviations. Let us decide that the primitives are

$$\bot\,(\text{falsity}),\ \wedge,\ \vee,\ \rightarrow,\ \forall,\ \exists;$$

$\neg F$ is an abbreviation for $F \rightarrow \bot$, $\top$ stands for $\bot \rightarrow \bot$, and $F \leftrightarrow G$ stands for $(F \rightarrow G) \wedge (G \rightarrow F)$.

Let $\mathbf{p}$ be a list of distinct predicate constants $p_1, \dots, p_n$. For any first-order formula F, by $\mathrm{SM}_{\mathbf{p}}[F]$ we denote the second-order formula

$$F \wedge \neg\exists\mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F^*(\mathbf{u})\big),$$

where $\mathbf{u}$ is a list of n distinct predicate variables $u_1, \dots, u_n$, and $F^*(\mathbf{u})$ is defined recursively:

- $p_i(\mathbf{t})^* = u_i(\mathbf{t})$ for any tuple $\mathbf{t}$ of terms;
- $F^* = F$ for any atomic formula F that does not contain members of $\mathbf{p}$[4];
- $(F \wedge G)^* = F^* \wedge G^*$;
- $(F \vee G)^* = F^* \vee G^*$;
- $(F \rightarrow G)^* = (F^* \rightarrow G^*) \wedge (F \rightarrow G)$;
- $(\forall x F)^* = \forall x F^*$;
- $(\exists x F)^* = \exists x F^*$.

If the list $\mathbf{p}$ is empty then we understand $\mathrm{SM}_{\mathbf{p}}[F]$ as F. We will drop the subscript in the symbol $\mathrm{SM}_{\mathbf{p}}$ when this does not lead to confusion.

For any sentence F, a $\mathbf{p}$-*stable* (or simply *stable*) *model* of F is an interpretation of the underlying signature that satisfies $\mathrm{SM}_{\mathbf{p}}[F]$.[5] Since the first conjunctive term of $\mathrm{SM}_{\mathbf{p}}[F]$ is F, it is clear that every stable model of F is a model of F.

Note that if we drop the second conjunctive term from the clause for implication in the definition of $F^*(\mathbf{u})$ then this formula will turn into $F(\mathbf{u})$, and $\mathrm{SM}[F]$ will turn into $\mathrm{CIRC}[F]$. It follows that for any sentence F that does not contain implication, $\mathrm{SM}[F]$ coincides with $\mathrm{CIRC}[F]$, and the stable models of F are identical to the minimal models of F.

In the next section we will see examples when these two formulas are equivalent to each other even though F does contain implication. We will see also that there are cases when minimal models are not stable, and when stable models are not minimal.

2.4. *Examples*

Example 1 *(continued).* Let F be formula (2). As noted above, $\mathrm{CIRC}_{pq}[F]$ is equivalent to (6). Consider the result of applying SM_{pq} to the same formula. Clearly $F^*(u, v)$ is

$$u(a,a) \wedge u(a,b) \wedge \forall xy\big(\big(u(x,y) \rightarrow v(x)\big) \wedge \big(p(x,y) \rightarrow q(x)\big)\big),$$

and $\mathrm{SM}_{pq}[F]$ is

$$\begin{aligned} &p(a,a) \wedge p(a,b) \wedge \forall xy\big(p(x,y) \rightarrow q(x)\big)\\ &\quad \wedge \neg\exists uv\big(\big((u,v) < (p,q)\big) \wedge u(a,a) \wedge u(a,b) \wedge \forall xy\big(\big(u(x,y) \rightarrow v(x)\big) \wedge \big(p(x,y) \rightarrow q(x)\big)\big)\big). \end{aligned}$$

In the presence of the conjunctive term $\forall xy(p(x,y) \rightarrow q(x))$ at the beginning of the formula, the conjunctive term $p(x,y) \rightarrow q(x)$ at the end can be dropped. This simplification turns $\mathrm{SM}_{pq}[F]$ into $\mathrm{CIRC}_{pq}[F]$. Consequently, $\mathrm{SM}_{pq}[F]$ is equivalent to (6) as well.

Remark 1. It is easy to see that, more generally, $\mathrm{SM}[F]$ is equivalent to $\mathrm{CIRC}[F]$ whenever F is a conjunction such that every conjunctive term

- does not contain implication, or
- is the universal closure of a formula $G \rightarrow H$ such that G and H do not contain implication.

Remark 2. The equivalence of $\mathrm{SM}_{pq}[F]$ to (6) in Example 1 can be established also in another way, without references to circumscription. In Sections 6.2 and 7.3 we will show how the theory of tight programs [4,2] can be extended to the

[4] This includes the case when F is $\bot$.

[5] The definition of a stable model used in the conference paper [6] and in related publications [13,16] is less general: it treats all predicate constants occurring in F as intensional. We will see that this additional degree of generality is convenient (Section 3.2) but not very essential (Section 4.1).

framework described in this paper, and we will see that the result of applying the operator SM can be often turned into a first-order formula using the process of program completion. This method can be applied, in particular, to formula (2).

Remark 3. According to the original definition of a stable model [9], the only stable model of program (1) is its minimal Herbrand model

$$\{p(a,a), p(a,b), q(a)\}. \tag{10}$$

This fact is in agreement with the result of the calculation in Example 1, in the sense that (10) is the only Herbrand interpretation satisfying (6). This is an instance of a general theorem about the relationship between the new, general definition of a stable model and the original definition, which is stated in Section 3 below.

Example 2 *(continued).* If F is (7) then $\mathrm{SM}_t[F]$ is equivalent to $\mathrm{CIRC}_t[F]$, according to Remark 1. Consequently, in the t-stable models of (7), t is the transitive closure of p. Similarly, if F is the conjunction of (7) and (8) then $\mathrm{SM}_{pt}[F]$ is equivalent to $\mathrm{CIRC}_{pt}[F]$ and consequently to (9).

It is clear from the definition of circumscription that if sentences F and G are equivalent to each other then the formulas $\mathrm{CIRC}[F]$ and $\mathrm{CIRC}[G]$ are equivalent to each other as well. The following example shows, on the other hand, that the operator SM, applied to two equivalent formulas, can produce formulas that are not equivalent to each other.

Example 3. Let us apply SM_p to $p(a)$ and to $\neg\neg p(a)$. (In logic programming notation the latter can be written as the constraint $\leftarrow$ *not* $p(a)$.) It is clear that $\mathrm{SM}_p[p(a)]$ equals $\mathrm{CIRC}_p[p(a)]$ and is equivalent to

$$\forall x\big(p(x) \leftrightarrow x = a\big).$$

On the other hand,

$$\begin{aligned}
\big(\neg\neg p(a)\big)^* &= \big(\big(p(a) \to \bot\big) \to \bot\big)^* \\
&= \big(\big(p(a) \to \bot\big)^* \to \bot\big) \wedge \big(\big(p(a) \to \bot\big) \to \bot\big) \\
&\leftrightarrow \neg\big(p(a) \to \bot\big)^* \wedge p(a) \\
&= \neg\big(\big(u(a) \to \bot\big) \wedge \big(p(a) \to \bot\big)\big) \wedge p(a) \\
&\leftrightarrow p(a),
\end{aligned}$$

and consequently

$$\begin{aligned}
\mathrm{SM}_p\big[\neg\neg p(a)\big] &\leftrightarrow \neg\neg p(a) \wedge \neg\exists u\big((u < p) \wedge p(a)\big) \\
&\leftrightarrow p(a) \wedge \neg\exists u(u < p) \\
&\leftrightarrow p(a) \wedge \forall x \neg p(x) \\
&\leftrightarrow \bot.
\end{aligned}$$

Thus some equivalent transformations do not preserve the class of stable models of a formula. We will return to this question in Section 5.1.

The following two examples show that sometimes SM is stronger than CIRC, and sometimes weaker.

Example 4. Let F be the formula

$$\forall x\big(\neg p(x) \to q(x)\big), \tag{11}$$

corresponding to the rule

$$q(x) \leftarrow \mathit{not}\ p(x).$$

The circumscription formula $\mathrm{CIRC}_{pq}[F]$ is equivalent to

$$\forall x\big(\neg p(x) \leftrightarrow q(x)\big).$$

On the other hand, using the fact that formula (11) is tight, we will show in Section 6 that $\mathrm{SM}_{pq}[F]$ can be written as

$$\forall x\big(\neg p(x) \wedge q(x)\big). \tag{12}$$

Thus $\mathrm{SM}_{pq}[F]$ is stronger than $\mathrm{CIRC}_{pq}[F]$. In any minimal model of (11), q is the negation of p; about the stable models of this formula we can say more: p is identically false, and q is identically true.

Example 5. Let F be formula (4), which represents a choice rule, as discussed above. Since this formula is logically valid, its p-minimal models are characterized by the condition

$$\forall x \neg p(x)$$

("p is empty"). Using the fact that formula (4) is tight, we will show in Section 6 that the p-stable models of (4) can be described, in accordance with the intuitive meaning of the choice construct, by the weaker condition

$$\forall x \big(p(x) \rightarrow q(x)\big) \tag{13}$$

("p is a subset of q").

3. Relation to other definitions of a stable model

In this section we relate the definition of a stable model in terms of the operator SM to the original definition of a stable model [9] and to the generalization of that definition proposed in [8].

3.1. Review of the 1988 and 2005 definitions

Recall that a *signature* is a set of object, function and predicate constants. A *term* of a signature σ is formed from object constants of σ and object variables using function constants of σ. We distinguish here between atoms and atomic formulas, as follows: an *atom* of a signature σ is an n-ary predicate constant followed by a list of n terms; *atomic formulas* of σ are atoms of σ, equalities between terms of σ, and the 0-place connective $\bot$. *First-order formulas* of σ are built from atomic formulas of σ using the binary propositional connectives and quantifiers listed at the beginning of Section 2.3. For any signature σ containing at least one object constant, an *Herbrand interpretation* of σ is an interpretation of σ such that (i) its universe is the set of ground terms of σ, and (ii) every ground term of σ represents itself. As usual, we identify an Herbrand interpretation with the set of ground atoms that are satisfied by it.

A *traditional program* of a signature σ is a set of formulas of the form

$$A_1 \wedge \cdots \wedge A_m \wedge \neg A_{m+1} \wedge \cdots \wedge \neg A_n \rightarrow A_{n+1} \tag{14}$$

($n \geqslant m \geqslant 0$), where each A_i is an atom of σ. If $n = 0$ then (14) is understood as A_1.

For any traditional program Π of a signature σ and any set X of ground atoms of σ, the *reduct* of Π relative to X is the set of formulas obtained from Π by

- replacing each formula from Π with all its ground instances, followed by
- removing all formulas (14) such that $\{A_{m+1}, \dots, A_n\} \cap X \neq \emptyset$, followed by
- removing the conjunctive terms $\neg A_{m+1}, \dots, \neg A_n$ from the antecedents of the remaining formulas.

The reduct of Π relative to X is a set of Horn clauses. If its least Herbrand model equals X then we say that X is a *stable model of Π in the sense of the 1988 definition* [9].

The definition from [8] is applicable to arbitrary sets of propositional formulas, and, if we include in it a grounding step, it will become applicable to arbitrary sets of quantifier-free formulas. For any set Π of quantifier-free formulas of a signature σ and any set X of ground atoms of σ, the *modified reduct* of Π relative to X is the set of formulas obtained from Π by

- replacing each formula from Π with all its ground instances, followed by
- replacing, in each formula F, all maximal subformulas of F that are not satisfied by X with $\bot$.

If X is a minimal (relative to set inclusion) Herbrand model of the modified reduct of Π relative to X then we say that X is a *stable model of Π in the sense of the 2005 definition* [8]. As shown in that paper, in application to any traditional program the 1988 and 2005 definitions are equivalent to each other.

Example 6. Signature σ consists of the object constants a, b and the unary predicate constants p, q, r; Π is

$$\big\{p(a), p(b), q(a), p(x) \wedge \neg q(x) \rightarrow r(x)\big\}; \tag{15}$$

X is

$$\big\{p(a), p(b), q(a), r(b)\big\}. \tag{16}$$

After grounding, Π becomes

$$\big\{p(a), p(b), q(a), p(a) \wedge \neg q(a) \rightarrow r(a), p(b) \wedge \neg q(b) \rightarrow r(b)\big\}.$$

The reduct of Π relative to X is

$$\{p(a), p(b), q(a), p(b) \to r(b)\}.$$

The least Herbrand model of the reduct equals X. Consequently X is a stable model of Π in the sense of the 1988 definition. The modified reduct of Π relative to X is

$$\{p(a), p(b), q(a), \bot \to \bot, p(b) \wedge \neg\bot \to r(b)\}.$$

Since X is a minimal model of the modified reduct, X is a stable model of Π in the sense of the 2005 definition.

3.2. Relation to the new definition

Theorem 1. *For any signature σ containing at least one object constant and finitely many predicate constants, any finite set Π of quantifier-free formulas of σ, and any Herbrand interpretation X of σ, the following conditions are equivalent:*

- *X is a stable model of Π in the sense of the 2005 definition;*
- *X is a* **p***-stable model of the conjunction of the universal closures of the formulas from Π, where* **p** *is the list of all predicate constants of σ.*

Corollary 1. *For any signature σ containing at least one object constant and finitely many predicate constants, any finite traditional program Π of σ, and any Herbrand interpretation X of σ, the following conditions are equivalent:*

- *X is a stable model of Π in the sense of the 1988 definition;*
- *X is a* **p***-stable model of the conjunction of the universal closures of the formulas from Π, where* **p** *is the list of all predicate constants of σ.*

Example 6 *(continued).* The result of applying the operator SM_{pqr} to the conjunction of the universal closures of formulas (15) can be rewritten, using the completion method described in Section 6 below, as

$$\begin{aligned}&\forall x\big(p(x) \leftrightarrow x = a \vee x = b\big)\\&\quad\wedge \forall x\big(q(x) \leftrightarrow x = a\big)\\&\quad\wedge \forall x\big(r(x) \leftrightarrow p(x) \wedge \neg q(x)\big).\end{aligned} \tag{17}$$

The stable model (16) of (15) is the only Herbrand model of this sentence.

In the statement of Theorem 1, the underlying signature is assumed to contain finitely many predicate constants, and Π is supposed to consist of finitely many formulas. (The result of grounding Π can be infinite though, if the signature contains function constants.) The theorem shows that under these conditions the new definition of a stable model is a generalization of the 2005 definition, and it is more general in three ways.

First, it is more general syntactically: it is applicable to formulas that contain both universal and existential quantifiers, such as the "aggregate formula" (5) or the formula $\exists x\, p(x)$ ("p is nonempty"). The result of applying the operator SM_p to the latter is the same as the result of applying the corresponding circumscription operator, and it is equivalent to

$$\exists x \forall y\big(p(y) \leftrightarrow x = y\big)$$

("p is a singleton").

Second, it is more general semantically: it is applicable to non-Herbrand interpretations. For instance, (17) has models in which some elements of the universe are not represented by any of the constants a, b. That formula has also models in which a and b represent the same element of the universe. In such a model, both p and q are singletons, and r is empty.

Third, it allows us to distinguish between intensional predicates and the other ("extensional") predicate symbols. This is often useful when we want to describe the intuitive meaning of a group of rules in a precise way. For instance, the claim that under the stable model semantics formula (7) expresses the concept of transitive closure is only valid if we treat p as extensional. (A way to express this claim without the use of extensional predicates is discussed in the next section.) See [7] for other uses of this distinction.

4. Properties of SM

4.1. Changing the set of intensional predicates

The theorem below shows that making the set of intensional predicates smaller can only make the result of applying the operator SM weaker, and that this can be compensated by adding "choice rules." For any predicate constant p, by $\mathit{Choice}(p)$ we denote the formula $\forall \mathbf{x}(p(\mathbf{x}) \vee \neg p(\mathbf{x}))$, where $\mathbf{x}$ is a list of distinct object variables. For any list $\mathbf{p}$ of predicate constants, $\mathit{Choice}(\mathbf{p})$ stands for the conjunction of the formulas $\mathit{Choice}(p)$ for all members p of $\mathbf{p}$.

Theorem 2. *For any first-order formula F and any disjoint lists $\mathbf{p}$, $\mathbf{q}$ of distinct predicate constants, the following formulas are logically valid*:

$$\mathrm{SM}_{\mathbf{pq}}[F] \to \mathrm{SM}_{\mathbf{p}}[F],$$
$$\mathrm{SM}_{\mathbf{pq}}\big[F \wedge \mathit{Choice}(\mathbf{q})\big] \leftrightarrow \mathrm{SM}_{\mathbf{p}}[F].$$

It follows that the class of $\mathbf{p}$-stable models of a sentence F contains the class of $\mathbf{pq}$-stable models of F and coincides with the class of $\mathbf{pq}$-stable models of $F \wedge \mathit{Choice}(\mathbf{q})$.

We have seen, for instance, that the condition "t is the transitive closure of p" can be expressed by applying SM_t to formula (7). By Theorem 2, it follows that the same condition can be expressed by applying SM_{pt} to the conjunction of (7) and $\forall x(p(x) \vee \neg p(x))$.

Thus the possibility of distinguishing between intensional and extensional predicates does not really make the concept of a stable model more general: instead of designating a group $\mathbf{q}$ of predicates as extensional, we can conjoin the formula with $\mathit{Choice}(\mathbf{q})$.

In the rest of the paper we will assume that a list $\mathbf{p}$ of distinct predicate constants is chosen, and its members will be referred to as intensional predicates. The predicate constants that do not belong to $\mathbf{p}$ will be called extensional predicates.

4.2. Constraints

In answer set programming, constraints—rules with the empty head, such as (3)—play an important role in view of the fact that adding a constraint to a program affects the set of its stable models in a particularly simple way: it eliminates the stable models that "violate" the constraint. The following theorem shows that sentences beginning with negation can be viewed as a counterpart of constraints in the new framework.

Theorem 3. *For any first-order formulas F and G, $\mathrm{SM}[F \wedge \neg G]$ is equivalent to $\mathrm{SM}[F] \wedge \neg G$.*

It follows that the stable models of a sentence of the form $F \wedge \neg G$ can be characterized as the stable models of F that satisfy $\neg G$.

For any predicate constant p, by $\mathit{False}(p)$ we denote the formula $\forall \mathbf{x} \neg p(\mathbf{x})$, where $\mathbf{x}$ is a list of distinct object variables. By $\mathit{False}(\mathbf{p})$ we denote the conjunction of the formulas $\mathit{False}(p)$ for all members p of $\mathbf{p}$.

Corollary 2. *For any first-order formula G, $\mathrm{SM}[\neg G]$ is equivalent to*

$$\neg G \wedge \mathit{False}(\mathbf{p}).$$

Indeed, if F is $\top$ then $\mathrm{SM}[F \wedge \neg G]$ is equivalent to $\mathrm{SM}[\neg G]$, and $\mathrm{SM}[F]$ is equivalent to $\mathit{False}(\mathbf{p})$.

In Section 5.1 we will show that $\neg G$ can be replaced in these two propositions by formulas of a more general syntactic form.

4.3. Trivial predicates

In traditional theory of stable models, the predicate constants that do not occur in the heads of rules are "trivial," in the sense that no atom containing such a predicate can belong to a stable model. Theorem 4 shows what form this idea takes in the new framework.

Theorem 4. *For any first-order formula F and any intensional predicate p, if every occurrence of p in F belongs to the antecedent of an implication then the formula*

$$\mathrm{SM}[F] \to \mathit{False}(p)$$

is logically valid.

Consequently, if every occurrence of p in a sentence F belongs to the antecedent of an implication then p is identically false in every stable model of F. For instance, the only occurrence of p in (7) is in the antecedent of an implication; consequently, in all p-stable models of (7) p is identically false.

Recall that an occurrence of a predicate constant (or any other expression) in a formula is called *positive* if the number of implications containing that occurrence in the antecedent is even, and *strictly positive* if that number is 0. The condition "every occurrence of p in F belongs to the antecedent of an implication" in the statement of the theorem can be also expressed by saying that F has no strictly positive occurrences of p.

5. Logic of here-and-there and strong equivalence

5.1. System $\mathbf{SQHT}^=$

As we saw in Section 2.4, two sentences that are equivalent to each other may have different stable models. Transformations of formulas that preserve the class of stable models were studied in [16], for the special case when all predicate constants are intensional. The results of that paper imply, in particular, that two sentences have the same stable models whenever they are *intuitionistically* equivalent.[6] We will see that the same conclusion holds in the more general framework proposed in this paper, with extensional predicates allowed.

Thus equivalent transformations that are sanctioned by intuitionistic logic play an important part in the study of stable models. In connection with Example 3 above we can note, for instance, that the "fact" $p(a)$ and the "constraint" $\neg\neg p(a)$ are equivalent classically, but not intuitionistically; this is what makes them essentially different under the stable model semantics. About formula (4), representing a choice rule, we can note that it is not provable in intuitionistic logic; this is what makes it nontrivial, as far as stable models are concerned.

The main result of [16] is actually about a class of equivalent transformations that contains more than what intuitionistic logic accepts. The "logic of here-and-there"[7] studied in that paper is intermediate between intuitionistic and classical logic. By $\mathbf{INT}^=$ we denote intuitionistic first-order predicate logic with the usual axioms for equality: $x = x$ and the schema

$$x = y \to \big(F(x) \to F(y)\big)$$

for every formula $F(x)$ such that y is substitutable for x in $F(x)$. System $\mathbf{SQHT}^=$ (for "*s*tatic *q*uantified logic of *h*ere-and-*t*here with equality") is obtained from $\mathbf{INT}^=$ by adding the axiom schemas

$$F \lor (F \to G) \lor \neg G$$

and

$$\exists x\big(F(x) \to \forall x F(x)\big),$$

and the axiom

$$x = y \lor x \neq y.$$

To illustrate the difference between intuitionistic logic and the logic of here-and-there, we can note that De Morgan's law

$$\neg(F \land G) \leftrightarrow \neg F \lor \neg G$$

and its first-order counterpart

$$\neg\forall x F(x) \leftrightarrow \exists x \neg F(x)$$

are not provable intuitionistically, but are provable in $\mathbf{SQHT}^=$.

If the equivalence between two sentences can be proved in $\mathbf{SQHT}^=$ then they have the same stable models. We can assert even more:

Theorem 5. *For any first-order formulas F and G, if the formula $F \leftrightarrow G$ is derivable in $\mathbf{SQHT}^=$ from the formulas Choice(q) for the extensional predicates q then* SM$[F]$ *is equivalent to* SM$[G]$.

For instance, it is easy to see that the equivalence between (4) and the formula

$$\forall x\big(p(x) \lor \neg p(x) \lor \neg q(x)\big) \tag{18}$$

is intuitionistically derivable from *Choice*(q). The p-stable models of (4) are the interpretations that interpret p as a subset of q (Section 2.4, Example 5). It follows that the p-stable models of (18) can be characterized in the same way.

Intermediate logics, such as $\mathbf{SQHT}^=$, differ from classical logic in that they do not endorse the law of double negation $\neg\neg F \leftrightarrow F$ in full generality. The following theorem identifies a class of cases when double negation elimination is admissible under the stable model semantics.

Theorem 6. *Let F' be the formula obtained from a first-order formula F by inserting $\neg\neg$ in front of a subformula G. If G has no strictly positive occurrences of intensional predicates then* SM$[F']$ *is equivalent to* SM$[F]$.

[6] See http://plato.stanford.edu/entries/logic-intuitionistic/ for an introduction to intuitionistic logic.

[7] This name is related to the fact that $\mathbf{SQHT}^=$ can be described by Kripke models with two worlds (see Appendix A.5.1), often called *Here* and *There*.

For instance, in a formula of the form

$$\forall xy(H \to x = y) \tag{19}$$

every occurrence of every predicate constant belongs to the antecedent of an implication. Consequently, inserting a double negation in front of (19) within any sentence will not affect the class of stable models no matter how the set of intensional predicates is chosen. (In the terminology of Section 5.2 below, this is a "strongly equivalent" transformation.)

From Theorem 6 we can conclude that Theorem 3 and Corollary 2 can be generalized: $\mathrm{SM}[F \wedge G]$ is equivalent to $\mathrm{SM}[F] \wedge G$, and $\mathrm{SM}[G]$ is equivalent to $G \wedge \mathit{False}(\mathbf{p})$, whenever G has no strictly positive occurrences of intensional predicates. For instance, $\mathrm{SM}[F \wedge \forall xy(H \to x = y)]$ is equivalent to $\mathrm{SM}[F] \wedge \forall xy(H \to x = y)$.

A generalization of Theorem 6 is presented in [7, Section 5].

5.2. Strong equivalence

About first-order formulas F and G we say that F is *strongly equivalent* to G if, for any formula H, any occurrence of F in H, and any list $\mathbf{p}$ of distinct predicate constants, $\mathrm{SM}_{\mathbf{p}}[H]$ is equivalent to $\mathrm{SM}_{\mathbf{p}}[H']$, where H' is obtained from H by replacing the occurrence of F by G. In this definition, H is allowed to contain object, function and predicate constants that do not occur in F, G; Theorem 7 below shows, however, that this is not essential. It shows also that in the definition of strong equivalence $\mathbf{p}$ can be taken to be the set $\mathbf{p}^{FG}$ of all predicate constants that occur in F or G, rather than an arbitrary set of predicate constants:

Theorem 7. *First-order formulas F and G are strongly equivalent to each other iff for any formula H such that every object, function or predicate constant occurring in H occurs in F or in G, and for any occurrence of F in H, $\mathrm{SM}_{\mathbf{p}^{FG}}[H]$ is equivalent to $\mathrm{SM}_{\mathbf{p}^{FG}}[H']$, where H' is obtained from H by replacing the occurrence of F by G.*

It is clear that if F is strongly equivalent to G then $\mathrm{SM}_{\mathbf{p}}[F]$ is equivalent to $\mathrm{SM}_{\mathbf{p}}[G]$ (take H to be F). In particular, if F is strongly equivalent to G then F is equivalent to G (take $\mathbf{p}$ to be empty).

Strong equivalence was originally defined, in somewhat different contexts, in [15] (for propositional rules with nested expressions, without extensional atoms, and assuming that F occurs in H as a conjunctive term) and in [16] (no free variables in F, G; no extensional predicates; F occurs in H as a conjunctive term). Properties of this relation are interesting from the perspective of ASP because they may allow us to simplify a part of a logic program without looking at the other parts. For instance, replacing the rule $p(x) \leftarrow x = a$ in any program with $p(a)$ does not affect the class of stable models, because the formula

$$\forall x\big(x = a \to p(x)\big) \tag{20}$$

is strongly equivalent to $p(a)$.

The main result of [16] can be extended to the new version of strong equivalence as follows:

Theorem 8. *First-order formulas F and G are strongly equivalent to each other iff formula $F \leftrightarrow G$ is provable in* $\mathbf{SQHT}^{=}$.

For instance, to prove that (20) is strongly equivalent to $p(a)$ we only need to observe that these formulas are intuitionistically equivalent.

The definition of strong equivalence can be generalized as follows. For any list $\mathbf{q}$ of predicate constants, we say that F *is strongly equivalent to G excluding* $\mathbf{q}$ if $F \wedge \mathit{Choice}(\mathbf{q})$ is strongly equivalent to $G \wedge \mathit{Choice}(\mathbf{q})$. It is immediate from Theorem 8 that F is strongly equivalent to G excluding $\mathbf{q}$ iff $F \leftrightarrow G$ is derivable in $\mathbf{SQHT}^{=}$ from the formula $\mathit{Choice}(\mathbf{q})$. Theorem 8 is the special case of this corollary when $\mathbf{q}$ is empty. Furthermore, it is clear from Theorem 5 that if F is strongly equivalent to G excluding $\mathbf{q}$ then $\mathrm{SM}_{\mathbf{p}}[F]$ is equivalent to $\mathrm{SM}_{\mathbf{p}}[G]$ for any $\mathbf{p}$ that is disjoint from $\mathbf{q}$.

An alternative characterization of strong equivalence, similar to the one proposed in [22] for the propositional case, refers to the formula $F^*(\mathbf{u})$ that was used in Section 2.3 to define the operator SM. In the statement of the theorem below, $\mathbf{p}^{FG}$ is again the list of all predicate constants that occur in F or G; $\mathbf{q}$ is a list of new, distinct predicate constants of the same length as $\mathbf{p}^{FG}$.

Theorem 9. *F is strongly equivalent to G iff the formula*

$$\big(\mathbf{q} \leqslant \mathbf{p}^{FG}\big) \to \big(F^*(\mathbf{q}) \leftrightarrow G^*(\mathbf{q})\big)$$

is logically valid.

For instance, we can prove that (20) is strongly equivalent to $p(a)$ by showing that the implication

$$(q \leqslant p) \to \Big(\forall x\big(\big(x = a \to q(x)\big) \wedge \big(x = a \to p(x)\big)\big) \leftrightarrow q(a)\Big)$$

is logically valid.

6. Completion

As indicated in Section 2.4, the process of completing a logic program, invented by Keith Clark [1], allows us in many cases to rewrite SM[F] as a first-order formula.

6.1. Clark normal form

The completion process involves a series of preliminary transformations followed by the main step—replacing implications by equivalences. For instance, completing program (1) can be described as follows. Step 1: in the representation (2) of the program in the syntax of first-order logic, we rewrite each conjunctive term as an implication with the consequent in a canonical form—a predicate constant followed by a list of distinct variables:

$$\begin{aligned}&\forall xy\big(x=a\wedge y=a\rightarrow p(x,y)\big)\wedge\forall xy\big(x=a\wedge y=b\rightarrow p(x,y)\big)\\&\quad\wedge\forall xy\big(p(x,y)\rightarrow q(x)\big).\end{aligned}$$

Step 2: we combine implications with the same predicate constant in the consequent into one:

$$\begin{aligned}&\forall xy\big(\big((x=a\wedge y=a)\vee(x=a\wedge y=b)\big)\rightarrow p(x,y)\big)\\&\quad\wedge\forall xy\big(p(x,y)\rightarrow q(x)\big).\end{aligned}$$

Step 3: we identify, in each implication, the variables that occur in its antecedent but do not occur in the consequent, and minimize the scopes of the corresponding quantifiers:

$$\begin{aligned}&\forall xy\big(\big((x=a\wedge y=a)\vee(x=a\wedge y=b)\big)\rightarrow p(x,y)\big)\\&\quad\wedge\forall x\big(\exists y p(x,y)\rightarrow q(x)\big).\end{aligned}\tag{21}$$

Step 4: we replace all implications by equivalences:

$$\begin{aligned}&\forall xy\big(p(x,y)\leftrightarrow(x=a\wedge y=a)\vee(x=a\wedge y=b)\big)\\&\quad\wedge\forall x\big(q(x)\leftrightarrow\exists y\ p(x,y)\big).\end{aligned}\tag{22}$$

Steps 1–3 are intuitionistically equivalent transformations, so that formula (21) has the same stable models as the formula (2) that we started with. Step 4 gives us in this case, and in many others, a first-order formula equivalent to the result of applying the operator SM.

This idea can be made precise using the following definitions. About a first-order formula we will say that it is in *Clark normal form* (relative to the list **p** of intensional predicates) if it is a conjunction of formulas of the form

$$\forall\mathbf{x}\big(G\rightarrow p(\mathbf{x})\big),\tag{23}$$

one for each intensional predicate p, where **x** is a list of distinct object variables. The *completion* of a formula F in Clark normal form, denoted by Comp[F], is obtained from it by replacing each conjunctive term (23) with

$$\forall\mathbf{x}\big(p(\mathbf{x})\leftrightarrow G\big).\tag{24}$$

For instance, (11) can be written in Clark normal form relative to pq as follows:

$$\forall x\big(\bot\rightarrow p(x)\big)\wedge\forall x\big(\neg p(x)\rightarrow q(x)\big).\tag{25}$$

The completion of this formula is

$$\forall x\big(p(x)\leftrightarrow\bot\big)\wedge\forall x\big(q(x)\leftrightarrow\neg p(x)\big).\tag{26}$$

Some formulas can be converted to Clark normal form by strongly equivalent transformations different from those described in [1]. For instance, formula (4) is strongly equivalent to

$$\forall x\big(q(x)\wedge\neg\neg p(x)\rightarrow p(x)\big),\tag{27}$$

because $F\vee\neg G$ is equivalent to $\neg\neg G\rightarrow F$ in $\mathbf{SQHT}^{=}$. Formula (27) is in Clark normal form relative to p. Its completion is

$$\forall x\big(p(x)\leftrightarrow q(x)\wedge\neg\neg p(x)\big),\tag{28}$$

or, equivalently, (13).

We are interested in the relationship between Comp[F] and SM[F]. In traditional theory of stable models, every stable model of a logic program is an Herbrand model of its completion; the converse, however, can be asserted only under some syntactic conditions of F, such as tightness [4,2]. Here is the counterpart of the first of these two facts in the new framework:

Theorem 10. *For any formula F in Clark normal form, the implication*

$$\mathrm{SM}[F] \to \mathrm{Comp}[F]$$

is logically valid.

To illustrate the fact that $\mathrm{Comp}[F]$ can be weaker than $\mathrm{SM}[F]$, consider the following formula, which is intuitionistically equivalent to (7):

$$\forall xy\big(p(x,y) \vee \exists z\big(t(x,z) \wedge t(z,y)\big) \to t(x,y)\big). \tag{29}$$

It is in Clark normal form, provided that t is taken to be the only intensional predicate. Its completion is weaker than the result of applying the operator SM_t to (7)—the latter, as we know, is not expressible in first-order logic.

6.2. Tight formulas

We will now define tightness for formulas in Clark normal form. In Section 7.3 this definition will be extended to arbitrary first-order formulas.

We say that an occurrence of a predicate constant in a formula is *negated* if it belongs to a subformula of the form $\neg F$ (that is, $F \to \bot$), and *nonnegated* otherwise.

For any formula F in Clark normal form, the *predicate dependency graph* of F is the directed graph that

- has all intensional predicates as its vertices, and
- has an edge from p to q if the antecedent G of the conjunctive term (23) of F with p in the consequent has a positive nonnegated occurrence of q.

We say that F is *tight* if the predicate dependency graph of F is acyclic.

For example, (21) is tight: its predicate dependency graph has only one edge, from q to p. Formulas (25) and (27) are tight as well: their predicate dependency graphs have no edges. (The antecedent in (27) has a positive occurrence of p, but that occurrence is negated.) On the other hand, (29) is not tight: the only edge of its predicate dependency graph is a self-loop.

Theorem 11. *For any tight formula F in Clark normal form,* $\mathrm{SM}[F]$ *is equivalent to the completion of F.*

In particular, the stable models of a tight sentence in Clark normal form can be characterized as models of its completion.

This theorem shows, for instance, that the result of applying the operator SM_{pq} to (2) is equivalent to formula (22). Since that formula can be equivalently rewritten as (6), we have justified the claim regarding Example 1 made in Section 2.4.

Similarly, the result of applying SM_{pq} to (11) is equivalent to (26). Since that formula can be equivalently rewritten as (12), we have justified the claim made there regarding Example 4.

Similarly, the result of applying SM_p to (4) is equivalent to (28). Since that formula can be equivalently rewritten as (13), we have justified the claim made there regarding Example 5.

These examples illustrate the process that sometimes allows us to rewrite $\mathrm{SM}[F]$ as a first-order formula:

- turn F into a tight formula in Clark normal form using strongly equivalent transformations, and
- form its completion (and simplify the result).

This process can be generalized in several ways. First, translating F into a tight formula F_1 in Clark normal form can employ transformations that are strongly equivalent excluding the extensional predicates; then the equivalence $F \leftrightarrow F_1$ will be derivable in **SQHT**$^=$ from the formulas *Choice*(q) for extensional predicates q, and that is enough to guarantee that $\mathrm{SM}[F]$ is equivalent to $\mathrm{SM}[F_1]$ (Theorem 5). Second, if we turned F into a conjunction of the form $F_1 \wedge \neg G$, where F_1 is in Clark normal form, then Theorem 3 can be used to "factor out" $\neg G$. Finally, if F is turned into a formula in Clark normal form that is not tight then in some cases tightness can be achieved by an additional transformation based on Theorem 6. For instance, the predicate dependency graph of a formula containing the conjunctive term

$$\forall x\Big(\big(\big(p(x) \to q(x)\big) \to r(x)\big) \to p(x)\Big)$$

has a self-loop at p. But if the predicate r is extensional then that term can be replaced with

$$\forall x\big(\neg\neg\big(\big(p(x) \to q(x)\big) \to r(x)\big) \to p(x)\big)$$

without changing the class of stable models. The self-loop is eliminated.

7. Pointwise stable models

The pointwise circumscription operator [18] is a modification of circumscription that reflects the idea of "pointwise minimality": it is impossible to make the minimized predicates stronger *by changing the truth value of exactly one of them at exactly one point*. In this section, we define a similar modification of the operator SM and show that it is closely related to the process of completion discussed above.[8]

7.1. Review of pointwise circumscription

The definition of pointwise circumscription uses the following notation. If p and q are predicate constants of the same arity k then $p \overset{1}{<} q$ stands for the formula

$$\exists \mathbf{x}\big(q(\mathbf{x}) \wedge \forall \mathbf{y}\big(p(\mathbf{y}) \leftrightarrow \big(q(\mathbf{y}) \wedge \mathbf{x} \neq \mathbf{y}\big)\big)\big),$$

where $\mathbf{x}$, $\mathbf{y}$ are disjoint tuples of distinct object variables $x_1, \dots, x_k$, $y_1, \dots, y_k$, and $\mathbf{x} \neq \mathbf{y}$ is shorthand for

$$\neg(x_1 = y_1 \wedge \cdots \wedge x_k = y_k).$$

The formula $p \overset{1}{<} q$ expresses that the extent of p can be obtained from the extent of q by removing one element. If $\mathbf{p}$ and $\mathbf{q}$ are tuples $p_1, \dots, p_n$ and $q_1, \dots, q_n$ of predicate constants then $\mathbf{p} \overset{1}{<} \mathbf{q}$ stands for the disjunction

$$\bigvee_{1 \leqslant i \leqslant n} \bigg(\big(p_i \overset{1}{<} q_i\big) \wedge \bigwedge_{1 \leqslant j \leqslant n,\ j \neq i} (p_j = q_j)\bigg),$$

and similarly for tuples of predicate variables.

Let $\mathbf{p}$ be a list of distinct predicate constants. The *pointwise circumscription operator with the minimized predicates* $\mathbf{p}$, denoted by $\mathrm{PCIRC}_{\mathbf{p}}$, is defined as follows: for any first-order formula F, $\mathrm{PCIRC}_{\mathbf{p}}[F]$ stands for

$$F \wedge \neg \exists \mathbf{u}\big(\big(\mathbf{u} \overset{1}{<} \mathbf{p}\big) \wedge F(\mathbf{u})\big),$$

where $\mathbf{u}$ and $F(\mathbf{u})$ are as in the definition of circumscription (Section 2.2). For any sentence F, a *pointwise* $\mathbf{p}$*-minimal model* of F is an interpretation of the underlying signature that satisfies $\mathrm{PCIRC}_{\mathbf{p}}[F]$.

It is clear that every minimal model is pointwise minimal. But the converse is not true. For instance, let F be $p(a) \leftrightarrow p(b)$. An interpretation that makes p true at two distinct points a, b and false in the rest of the universe is not minimal—it can be "improved" by making p identically false. But it is pointwise minimal, because changing the value of p at one of the points a, b would not produce a model of F.

Unlike CIRC[F], the pointwise circumscription formula PCIRC[F] can be equivalently rewritten without second-order quantifiers. We will describe this process in terms of *predicate expressions* $\lambda \mathbf{x} F(\mathbf{x})$, where $\mathbf{x}$ is a list of distinct object variables, and $F(\mathbf{x})$ is a formula. For any formula $H(u)$, where u is a predicate variable, by $H(\lambda \mathbf{x} F(\mathbf{x}))$ we denote the formula obtained from $H(u)$ by replacing each atomic subformula of the form $u(\mathbf{t})$, where $\mathbf{t}$ is a tuple of terms, with $F(\mathbf{t})$. For instance, if $H(u)$ is $u(a) \vee u(b)$ then $H(\lambda x \neg p(x))$ stands for $\neg p(a) \vee \neg p(b)$.

For any predicate variable v and any formula $H(v)$, by $H^{(1)}_v(v)$ we denote the formula

$$\exists \mathbf{x}\big(v(\mathbf{x}) \wedge H\big(\lambda \mathbf{y}\big(v(\mathbf{y}) \wedge \mathbf{x} \neq \mathbf{y}\big)\big)\big),$$

where $\mathbf{x}$ and $\mathbf{y}$ are disjoint lists of distinct variables. It is easy to see that this formula is equivalent to

$$\exists u\big(\big(u \overset{1}{<} v\big) \wedge H(u)\big).$$

Indeed,

$$\begin{aligned}
&\exists u\big(\big(u \overset{1}{<} v\big) \wedge H(u)\big)\\
&\quad = \exists u\big(\exists \mathbf{x}\big(v(\mathbf{x}) \wedge \forall \mathbf{y}\big(u(\mathbf{y}) \leftrightarrow \big(v(\mathbf{y}) \wedge \mathbf{x} \neq \mathbf{y}\big)\big)\big) \wedge H(u)\big)\\
&\quad \leftrightarrow \exists u \exists \mathbf{x}\big(v(\mathbf{x}) \wedge \forall \mathbf{y}\big(u(\mathbf{y}) \leftrightarrow \big(v(\mathbf{y}) \wedge \mathbf{x} \neq \mathbf{y}\big)\big) \wedge H(u)\big)\\
&\quad \leftrightarrow \exists \mathbf{x}\big(v(\mathbf{x}) \wedge \exists u\big(\forall \mathbf{y}\big(u(\mathbf{y}) \leftrightarrow \big(v(\mathbf{y}) \wedge \mathbf{x} \neq \mathbf{y}\big)\big) \wedge H(u)\big)\big)\\
&\quad \leftrightarrow H^{(1)}_v(v).
\end{aligned}$$

To generalize this construction to tuples of distinct predicate variables, we define $H^{(1)}_{v_1 \cdots v_n}$ as shorthand for

$$H^{(1)}_{v_1} \vee \cdots \vee H^{(1)}_{v_n}.$$

[8] In propositional case, an analogy between pointwise circumscription and completion was noted in [12].

The following calculation shows that $H^{(1)}_{\mathbf{v}}(\mathbf{v})$ is equivalent to

$$\exists\mathbf{u}\big((\mathbf{u} \overset{1}{<} \mathbf{v}) \wedge H(\mathbf{u})\big)$$

(to simplify notation, we assume that $n = 2$):

$$\begin{aligned}
&\exists u_1u_2\big(\big((u_1,u_2) \overset{1}{<} (v_1,v_2)\big) \wedge H(u_1,u_2)\big)\\
&\quad\leftrightarrow \exists u_1u_2\big(\big(\big((u_1 \overset{1}{<} v_1) \wedge (u_2 = v_2)\big) \vee \big((u_1 = v_1) \wedge (u_2 \overset{1}{<} v_2)\big)\big) \wedge H(u_1,u_2)\big)\\
&\quad\leftrightarrow \exists u_1u_2\big(\big((u_1 \overset{1}{<} v_1) \wedge (u_2 = v_2)\big) \wedge H(u_1,u_2)\big)\\
&\qquad \vee \exists u_1u_2\big(\big((u_1 = v_1) \wedge (u_2 \overset{1}{<} v_2)\big) \wedge H(u_1,u_2)\big)\\
&\quad\leftrightarrow \exists u_1\big((u_1 \overset{1}{<} v_1) \wedge H(u_1,v_2)\big) \vee \exists u_2\big((u_2 \overset{1}{<} v_2) \wedge H(v_1,u_2)\big)\\
&\quad\leftrightarrow H^{(1)}_{v_1}(v_1,v_2) \vee H^{(1)}_{v_2}(v_1,v_2)\\
&\quad= H^{(1)}_{v_1v_2}(v_1,v_2).
\end{aligned}$$

Consequently, $\mathrm{PCIRC}_{\mathbf{p}}[F]$ is equivalent to

$$F \wedge \neg F^{(1)}_{\mathbf{u}}(\mathbf{p}),$$

which is a first-order formula. For instance, this translation turns

$$\mathrm{PCIRC}_p\big[p(a) \leftrightarrow p(b)\big]$$

into the first-order formula

$$\big(p(a) \leftrightarrow p(b)\big) \wedge \neg\exists x\big(p(x) \wedge \big(\big(p(a) \wedge x \neq a\big) \leftrightarrow \big(p(b) \wedge x \neq b\big)\big)\big),$$

which can be further rewritten as

$$\forall x\neg p(x) \vee \big(a \neq b \wedge \forall x\big(p(x) \leftrightarrow x = a \vee x = b\big)\big).$$

7.2. *Operator* PSM

The *pointwise stable model operator with the intensional predicates* $\mathbf{p}$, denoted by $\mathrm{PSM}_{\mathbf{p}}$, is defined as follows: for any first-order formula F, $\mathrm{PSM}_{\mathbf{p}}[F]$ stands for

$$F \wedge \neg\exists\mathbf{u}\big((\mathbf{u} \overset{1}{<} \mathbf{p}) \wedge F^*(\mathbf{u})\big),$$

where $\mathbf{u}$ and $F^*(\mathbf{u})$ are as in the definition of the stable model operator (Section 2.3). For any sentence F, a *pointwise* $\mathbf{p}$*-stable model* of F is an interpretation of the underlying signature that satisfies $\mathrm{PSM}_{\mathbf{p}}[F]$.

Every stable model is pointwise stable, but the converse is generally not true. Furthermore, $\mathrm{PSM}[F]$ is equivalent to the first-order formula

$$F \wedge \neg\big(F^*\big)^{(1)}_{\mathbf{u}}(\mathbf{p}).$$

We see that there is a similarity between properties of PSM and properties of completion. Indeed, for any sentence F in Clark normal form, every stable model of F satisfies the completion of F (Theorem 10), but the converse is generally not true; the completion of F is a first-order formula. The difference is, of course, that the definition of PSM is more general—it is not limited to sentences in Clark normal form.

Theorem 12(b) below shows that this is more than a similarity: PSM can be viewed as a generalization of completion.

About a sentence in Clark normal form we say that it is *pure* if, for each of its conjunctive terms (23), G has no strictly positive occurrences of p. For instance, every tight sentence is pure. Any formula in Clark normal form can be made pure using auxiliary predicates. For instance, formula (29) is not pure, but we can make it pure using the auxiliary predicate t', "synonymous" with t:

$$\forall xy\big(p(x,y) \vee \exists z\big(t(x,z) \wedge t(z,y)\big) \rightarrow t'(x,y)\big) \wedge \forall xy\big(t'(x,y) \rightarrow t(x,y)\big).$$

Theorem 12. *For any formula F in Clark normal form,* (a) *the implication*

$$\mathrm{PSM}[F] \rightarrow \mathrm{Comp}[F]$$

is logically valid; (b) *if F is pure then* $\mathrm{PSM}[F]$ *is equivalent to* $\mathrm{Comp}[F]$.

When applied to a formula in Clark normal form that is not pure, PSM provides, generally, a better approximation to SM than the completion operator.

7.3. Tight formulas revisited

As the final comment on the concept of a pointwise stable model, we will show how to extend the tightness condition from formulas in Clark normal form to arbitrary formulas so that a counterpart of Theorem 11 will hold: for a tight formula F, SM$[F]$ will be equivalent to PSM$[F]$ (and consequently equivalent to a first-order formula).

A *rule* of a first-order formula F is a strictly positive occurrence of an implication in F. For instance, the only rule of (2) is $p(x, y) \to q(x)$. (Note that the first two conjunctive terms of (2) are not rules, according to our definition.) If F is a formula in Clark normal form then its rules are the implications $G \to p(\mathbf{x})$ from its conjunctive terms (23). The rules of the formula

$$\big(p(x) \to \big(q(x) \to r(x)\big)\big) \vee \big(\big(p(y) \to q(y)\big) \to r(y)\big)$$

are

$$p(x) \to \big(q(x) \to r(x)\big), \quad q(x) \to r(x), \quad \big(p(y) \to q(y)\big) \to r(y).$$

For any first-order formula F, the *predicate dependency graph* of F (relative to the list $\mathbf{p}$ of intensional predicates) is the directed graph that

- has all intensional predicates as its vertices, and
- has an edge from p to q if, for some rule $G \to H$ of F,
  - · p has a strictly positive occurrence in H, and
  - · q has a positive nonnegated occurrence in G.

We say that F is *tight* (relative to $\mathbf{p}$) if its predicate dependency graph is acyclic.

In application to formulas in Clark normal form, the new definition of tightness is equivalent to the definition from Section 6.2. But it allows us to talk, for instance, about the predicate dependency graph of formula (2) itself, without converting it to Clark normal form, and say that (2) itself is tight. Incidentally, this formula and its normal form (21) have the same predicate dependency graph, and this is a general phenomenon: strongly equivalent transformations involved in converting a sentence to its Clark normal form do not usually change its predicate dependency graph, and consequently do not affect its tightness.

Theorem 13. *For any tight formula F,* PSM$[F]$ *is equivalent to* SM$[F]$.

Corollary 3. *For any tight formula F,* SM$[F]$ *is equivalent to a first-order formula.*

8. Strong negation

Some applications of answer set programming are facilitated by the use of a second kind of negation, called “strong” or “classical” [10].

Strong negation can be incorporated in the framework of this paper as follows. We distinguish between intensional predicates of two kinds, *positive* and *negative*, and assume that each negative intensional predicate has the form $\sim p$, where p is a positive intensional predicate. Under this approach to strong negation, the symbol $\sim$ is, syntactically, not a connective; it occurs within atomic formulas. An interpretation of the underlying signature is *coherent* if the extent of every negative predicate $\sim p$ in it is disjoint from the extent of the corresponding positive predicate p. In other words, an interpretation is coherent if it satisfies the formula

$$\neg\exists\mathbf{x}\big(p(\mathbf{x}) \wedge \sim p(\mathbf{x})\big), \tag{30}$$

where $\mathbf{x}$ is a list of distinct object variables, for each negative predicate $\sim p$.

By Theorem 3, the coherent stable models of a sentence F can be characterized as the stable model of the conjunction of F with all formulas (30).

Strong negation allows us to distinguish between two kinds of exceptions to defaults: when the default is not applicable, so that the property asserted by the default is not guaranteed to hold, and when we know that the property indeed does not hold. For instance, the formula

$$\forall x\big(\neg ab(x) \to p(x)\big) \wedge ab(c_1) \wedge ab(c_2) \wedge \sim p(c_2) \tag{31}$$

employs the “abnormality predicate” ab to express that

- by default, any object is presumed to have property p,
- this default is applicable neither to c_1 nor to c_2,
- c_2 does not have property p.

The completion method (Section 6) can be used to characterize the stable models of this formula, with all predicate constants treated as intensional, by a first-order formula:

$$\begin{aligned}&\forall x\big(ab(x) \leftrightarrow x = c_1 \vee x = c_2\big) \wedge\\ &\forall x\big(p(x) \leftrightarrow x \neq c_1 \wedge x \neq c_2\big) \wedge\\ &\forall x\big({\sim}p(x) \leftrightarrow x = c_2\big).\end{aligned}$$

According to this formula, all objects other than c_1 and c_2 have property p (line 2); as to c_1 and c_2, it is not known whether the former has property p, but the latter certainly doesn't (line 3).

All stable models of (31) are coherent. But this will change if we drop the conjunctive term $ab(c_2)$ from that formula, that is to say, if we assert ${\sim}p(c_2)$ but do not restrict accordingly the default that leads to the opposite conclusion. The completion formula will turn then into

$$\begin{aligned}&\forall x\big(ab(x) \leftrightarrow x = c_1\big) \wedge\\ &\forall x\big(p(x) \leftrightarrow x \neq c_1\big) \wedge\\ &\forall x\big({\sim}p(x) \leftrightarrow x = c_2\big).\end{aligned}$$

This sentence has no coherent models satisfying $c_1 \neq c_2$.

9. Related work

Propositional equilibrium logic [32] extends the stable model semantics from traditional programs to propositional formulas, and the definition of a stable model for first-order sentences proposed in this paper is a natural next step. It is closely related to the extension of equilibrium logic to first-order formulas described in Appendices A.5.1 and A.5.2.

Theorem 5 from [24] relates stable models of traditional programs to circumscription using a translation that introduces auxiliary predicate constants. Our approach to stable models is closer, however, to two more recent publications: [31], which shows how to express the semantics of propositional equilibrium logic by quantified Boolean formulas, and [23], which translates equilibrium logic into the logic of knowledge and justified assumptions from [21]. (An extended version of [23] is published in this issue.)

Non-Herbrand stable models, at least for traditional programs, can be defined on the basis of several characterizations of the stable model semantics proposed earlier, including [24,35,20].

Extensional predicates are similar to input predicates in the sense of [30].

10. Conclusion

The approach to stable models proposed in this paper is more general than the traditional definition because it is applicable to syntactically complex formulas, because it covers non-Herbrand models, and because it allows us to distinguish between intensional and extensional predicates. Syntactically complex formulas are useful in the context of the stable model semantics in view of their relation to aggregates. Non-Herbrand models are related to the use of arithmetic functions in logic programs. Extensional predicates provide a useful technical device, as discussed in [7].

Acknowledgements

We are grateful to Pedro Cabalar, Martin Gebser, Michael Gelfond, Fangzhen Lin, David Pearce and Hudson Turner for useful discussions related to the topic of this paper. This work was partially supported by the National Science Foundation under Grants IIS-0712113, IIS-0839821, and IIS-0916116.

Appendix A. Proofs of theorems

A.1. Proof of Theorem 1

Given a formula F without variables and a set X of ground atoms, by F^X we denote the modified reduct of F relative to X (Section 3.1), that is, the result of replacing all maximal subformulas of F that are not satisfied by X with $\bot$. Similar notation will be used for sets of ground formulas.

Lemma 1. *(See [5, Lemma 22].)* $X \models F^X$ *iff* $X \models F$.

Proof. Immediate from the definition of F^X. □

Lemma 2. *(See [5, Lemma 23].)* (a) $(F \wedge G)^X$ *is equivalent to* $F^X \wedge G^X$; (b) $(F \vee G)^X$ *is equivalent to* $F^X \vee G^X$.

Proof. (a) If X satisfies $F \wedge G$ then the formulas $(F \wedge G)^X$ and $F^X \wedge G^X$ are equal to each other; otherwise, each of them is equivalent to $\bot$. (b) Similar. $\square$

The following lemma is a key to the proof of Theorem 1. It relates the modified reduct operator to the operator $F \mapsto F^*(\mathbf{u})$ introduced in Section 2.3. In the statement of the lemma,

- $H(\mathbf{x})$ is a quantifier-free formula, $\mathbf{x}$ is the list of all its variables, and $\mathbf{t}$ is a list of ground terms of the same length as $\mathbf{x}$;
- $\mathbf{p}$ is the list of all predicate constants occurring in $H(\mathbf{x})$, and $\mathbf{q}$ is a list of new predicate constants of the same length as $\mathbf{p}$;
- X is a set of ground atoms that contain a predicate constant from $\mathbf{p}$, Y is a subset of X, and $Y^{\mathbf{p}}_{\mathbf{q}}$ is the set of ground atoms obtained from Y by substituting the members of $\mathbf{q}$ for the corresponding members of $\mathbf{p}$.

Lemma 3. *The Herbrand interpretation Y satisfies $H(\mathbf{t})^X$ iff the Herbrand interpretation $X \cup Y^{\mathbf{p}}_{\mathbf{q}}$ satisfies the sentence $H^*(\mathbf{q}, \mathbf{t})$ obtained from $H^*(\mathbf{u}, \mathbf{x})$ by substituting $\mathbf{q}$ for the predicate variables $\mathbf{u}$ and $\mathbf{t}$ for the object variables $\mathbf{x}$.*

Proof. By induction on H.

Case 1: $H(\mathbf{x})$ has the form $t_1(\mathbf{x}) = t_2(\mathbf{x})$. Then $H^*(\mathbf{q}, \mathbf{t})$ is $t_1(\mathbf{t}) = t_2(\mathbf{t})$; $X \cup Y^{\mathbf{p}}_{\mathbf{q}}$ satisfies this sentence iff $t_1(\mathbf{t})$ equals $t_2(\mathbf{t})$. On the other hand, $H(\mathbf{t})^X$ is $t_1(\mathbf{t}) = t_2(\mathbf{t})$ if $t_1(\mathbf{t})$ equals $t_2(\mathbf{t})$, and $\bot$ otherwise.

Case 2: $H(\mathbf{x})$ has the form $p(\mathbf{t}'(\mathbf{x}))$, where $\mathbf{t}'(\mathbf{x})$ is a tuple of terms. Then $H^*(\mathbf{q}, \mathbf{t})$ is $q(\mathbf{t}'(\mathbf{t}))$, where q is the member of $\mathbf{q}$ corresponding to the member p of $\mathbf{p}$; $X \cup Y^{\mathbf{p}}_{\mathbf{q}}$ satisfies this sentence iff $p(\mathbf{t}'(\mathbf{t}))$ belongs to Y. On the other hand, $H(\mathbf{t})^X$ is $p(\mathbf{t}'(\mathbf{t}))$ if this atom belongs to X, and $\bot$ otherwise. Since $Y \subseteq X$, Y satisfies $H(\mathbf{t})^X$ iff $p(\mathbf{t}'(\mathbf{t}))$ belongs to Y.

Case 3: $H(\mathbf{x})$ is $\bot$; trivial.

Case 4: $H(\mathbf{x})$ is a conjunction or a disjunction; use Lemma 2.

Case 5: $H(\mathbf{x})$ is $H_1(\mathbf{x}) \to H_2(\mathbf{x})$. Then $H^*(\mathbf{q}, \mathbf{t})$ is

$$H(\mathbf{t}) \wedge \big(H_1^*(\mathbf{q}, \mathbf{t}) \to H_2^*(\mathbf{q}, \mathbf{t})\big). \tag{A.1}$$

Case 5.1: $X \models H(\mathbf{t})$. Then the Herbrand interpretation $X \cup Y^{\mathbf{p}}_{\mathbf{q}}$ satisfies the conjunction (A.1) iff it satisfies its second term $H_1^*(\mathbf{q}, \mathbf{t}) \to H_2^*(\mathbf{q}, \mathbf{t})$. On the other hand, $H(\mathbf{t})^X$ is in this case $H_1(\mathbf{t})^X \to H_2(\mathbf{t})^X$, and it remains to apply the induction hypothesis.

Case 5.2: $X \not\models H(\mathbf{t})$. Then $X \cup Y^{\mathbf{p}}_{\mathbf{q}}$ does not satisfy (A.1), and $H(\mathbf{t})^X$ is $\bot$. $\square$

Theorem 1. *For any signature σ containing at least one object constant and finitely many predicate constants, any finite set Π of quantifier-free formulas of σ, and any Herbrand interpretation X of σ, the following conditions are equivalent:*

- *X is a stable model of Π in the sense of the 2005 definition;*
- *X is a $\mathbf{p}$-stable model of the conjunction of the universal closures of the formulas from Π, where $\mathbf{p}$ is the list of all predicate constants of σ.*

Proof. Let Π_g be the set of all ground instances of the formulas from Π, let $\mathbf{x}$ be the list of all variables occurring in Π, and let $F(\mathbf{x})$ be the conjunction of all formulas from Π. In view of Lemma 1, X is a stable model of Π in the sense of the 2005 definition iff

(i) X satisfies Π_g, and
(ii) no proper subset Y of X satisfies Π_g^X.

On the other hand, X is a $\mathbf{p}$-stable model of $\forall\mathbf{x}F(\mathbf{x})$ iff

(i′) X satisfies $\forall\mathbf{x}F(\mathbf{x})$, and
(ii′) X does not satisfy $\exists\mathbf{u}((\mathbf{u} < \mathbf{p}) \wedge \forall\mathbf{x}F^*(\mathbf{u}, \mathbf{x}))$.

It is clear that (i) is equivalent to (i′). By Lemma 2(a), condition (ii) can be reformulated as follows: no proper subset Y of X satisfies all of the formulas $(F(\mathbf{t}))^X$ for arbitrary tuples $\mathbf{t}$ of ground terms. Condition (ii′) can be reformulated in terms of a tuple of new predicate constants $\mathbf{q}$: there is no proper subset Y of X such that, for every tuple $\mathbf{t}$ of ground terms, $X \cup Y^{\mathbf{p}}_{\mathbf{q}}$ satisfies $F^*(\mathbf{q}, \mathbf{t})$. By Lemma 3, it follows that (ii) is equivalent to (ii′). $\square$

A.2. Proof of Theorem 2

Lemma 4. *For any list $\mathbf{p}$ of predicate constants, Choice$(\mathbf{p})^*(\mathbf{u})$ is equivalent to $\mathbf{p} \leqslant \mathbf{u}$.*

Proof. $(\forall \mathbf{x}(p(\mathbf{x}) \vee \neg p(\mathbf{x})))^*$ is

$$\forall \mathbf{x}\big(u(\mathbf{x}) \vee \big(\neg u(\mathbf{x}) \wedge \neg p(\mathbf{x})\big)\big);$$

$p \leqslant u$ is

$$\forall \mathbf{x}\big(p(\mathbf{x}) \rightarrow u(\mathbf{x})\big). \qquad \square$$

Theorem 2. *For any first-order formula F and any disjoint lists* $\mathbf{p}$, $\mathbf{q}$ *of distinct predicate constants, the following formulas are logically valid*:

$$\mathrm{SM}_{\mathbf{pq}}[F] \rightarrow \mathrm{SM}_{\mathbf{p}}[F],$$
$$\mathrm{SM}_{\mathbf{pq}}\big[F \wedge \mathit{Choice}(\mathbf{q})\big] \leftrightarrow \mathrm{SM}_{\mathbf{p}}[F].$$

The proof is not long, but there is a notational difficulty that we need to overcome before we can present it. The notation $F^*(\mathbf{u})$ introduced in Section 2.3 does not take into account the fact that the construction of this formula depends on the choice of the list $\mathbf{p}$ of intensional predicates. Since the dependence on $\mathbf{p}$ is essential in the proof of Theorem 2, we use here the more elaborate notation $F^{*[\mathbf{p}]}(\mathbf{u})$. For instance, if F is $p(x) \wedge q(x)$ then

$$F^{*[p]}(u) \quad \text{is } u(x) \wedge q(x),$$
$$F^{*[pq]}(u, v) \quad \text{is } u(x) \wedge v(x).$$

It is easy to verify by induction on F that for any disjoint lists $\mathbf{p}$, $\mathbf{q}$ of distinct predicate constants,

$$F^{*[\mathbf{p}]}(\mathbf{u}) = F^{*[\mathbf{pq}]}(\mathbf{u}, \mathbf{q}). \tag{A.2}$$

Proof of Theorem 2. (i) In the notation introduced above, $\mathrm{SM}_{\mathbf{p}}[F]$ is

$$F \wedge \neg \exists \mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F^{*[\mathbf{p}]}(\mathbf{u})\big).$$

By (A.2), this formula can be written also as

$$F \wedge \neg \exists \mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F^{*[\mathbf{pq}]}(\mathbf{u}, \mathbf{q})\big),$$

which is equivalent to

$$F \wedge \neg \exists \mathbf{u}\big(\big((\mathbf{u}, \mathbf{q}) < (\mathbf{p}, \mathbf{q})\big) \wedge F^{*[\mathbf{pq}]}(\mathbf{u}, \mathbf{q})\big).$$

On the other hand, $\mathrm{SM}_{\mathbf{pq}}[F]$ is

$$F \wedge \neg \exists \mathbf{uv}\big(\big((\mathbf{u}, \mathbf{v}) < (\mathbf{p}, \mathbf{q})\big) \wedge F^{*[\mathbf{pq}]}(\mathbf{u}, \mathbf{v})\big).$$

To prove (ii), note that, by (A.2) and Lemma 4, the formula

$$\exists \mathbf{uv}\big(\big((\mathbf{u}, \mathbf{v}) < (\mathbf{p}, \mathbf{q})\big) \wedge F^{*[\mathbf{pq}]}(\mathbf{u}, \mathbf{v}) \wedge \mathit{Choice}(\mathbf{q})^{*[\mathbf{pq}]}(\mathbf{u}, \mathbf{v})\big)$$

is equivalent to

$$\exists \mathbf{uv}\big(\big((\mathbf{u}, \mathbf{v}) < (\mathbf{p}, \mathbf{q})\big) \wedge F^{*[\mathbf{pq}]}(\mathbf{u}, \mathbf{v}) \wedge (\mathbf{q} = \mathbf{v})\big).$$

It follows that it can be also equivalently rewritten as

$$\exists \mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F^{*[\mathbf{pq}]}(\mathbf{u}, \mathbf{q})\big).$$

By (A.2), the last formula can be represented as

$$\exists \mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F^{*[\mathbf{p}]}(\mathbf{u})\big). \qquad \square$$

A.3. Proof of Theorem 3

Lemma 5. *The formula*

$$(\mathbf{u} \leqslant \mathbf{p}) \wedge F^*(\mathbf{u}) \rightarrow F$$

is logically valid.

Proof. By induction on F. $\square$

Lemma 6. *Formula*

$$\mathbf{u} \leqslant \mathbf{p} \to \big((\neg F)^*(\mathbf{u}) \leftrightarrow \neg F\big)$$

is logically valid.

Proof. Immediate from Lemma 5. □

Theorem 3. *For any first-order formulas F and G,* SM$[F \wedge \neg G]$ *is equivalent to* SM$[F] \wedge \neg G$.

Proof. By Lemma 6,

$$\begin{aligned}
\mathrm{SM}_{\mathbf{p}}[F \wedge \neg G] &= F \wedge \neg G \wedge \neg\exists\mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge (F \wedge \neg G)^*(\mathbf{u})\big)\\
&\Leftrightarrow F \wedge \neg G \wedge \neg\exists\mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F^*(\mathbf{u}) \wedge \neg G\big)\\
&\Leftrightarrow F \wedge \neg\exists\mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F^*(\mathbf{u})\big) \wedge \neg G\\
&= \mathrm{SM}_{\mathbf{p}}[F] \wedge \neg G. \qquad \square
\end{aligned}$$

A.4. Proof of Theorem 4

Lemma 7. *Assume that the set of intensional predicates is divided into two parts* $\mathbf{p}$, $\mathbf{q}$ *so that every occurrence of every predicate constant from* $\mathbf{p}$ *in F belongs to the antecedent of an implication. Then the formula*

$$(\mathbf{u} \leqslant \mathbf{p}) \to \big(F^*(\mathbf{u}, \mathbf{q}) \leftrightarrow F\big)$$

is logically valid.

(Lemma 6 is the special case of this assertion when F has the form $\neg G$, and $\mathbf{q}$ is empty.)

Proof. By induction on F. We will consider the case when F is $G \to H$; the other cases are straightforward. Assume $\mathbf{u} \leqslant \mathbf{p}$. By Lemma 5, it follows that $G^*(\mathbf{u}, \mathbf{q}) \to G$; by the induction hypothesis, $H^*(\mathbf{u}, \mathbf{q}) \leftrightarrow H$. Consequently

$$\begin{aligned}
F^*(\mathbf{u}, \mathbf{q}) &= \big(G^*(\mathbf{u}, \mathbf{q}) \to H^*(\mathbf{u}, \mathbf{q})\big) \wedge (G \to H)\\
&\Leftrightarrow \big(G^*(\mathbf{u}, \mathbf{q}) \to H\big) \wedge (G \to H)\\
&\Leftrightarrow \big(G^*(\mathbf{u}, \mathbf{q}) \vee G\big) \to H\\
&\Leftrightarrow G \to H\\
&= F. \qquad \square
\end{aligned}$$

Theorem 4. *For any first-order formula F and any intensional predicate p, if every occurrence of p in F belongs to the antecedent of an implication then the formula*

$$\mathrm{SM}[F] \to \mathit{False}(p)$$

is logically valid.

Proof. Let $\mathbf{q}$ be the set of all intensional predicates other than p. The formula to be proved can be written as

$$F \wedge \neg \mathit{False}(p) \to \exists u\mathbf{v}\big(\big((u, \mathbf{v}) < (p, \mathbf{q})\big) \wedge F^*(u, \mathbf{v})\big). \tag{A.3}$$

Assume $F \wedge \neg \mathit{False}(p)$, and take u such that $u < p$. By Lemma 7, it follows that $F^*(u, \mathbf{q})$. Hence

$$\big((u, \mathbf{q}) < (p, \mathbf{q})\big) \wedge F^*(u, \mathbf{q}),$$

which implies the consequent of (A.3). □

A.5. Proofs of Theorems 5–8

It is convenient to prove Theorems 7 and 8 before Theorems 5 and 6. As a preliminary step, in Lemma 9 below we extend the work on the relationship between stable models and propositional equilibrium logic described in [32] to the first-order case.

A.5.1. Kripke semantics for $\mathbf{SQHT}^=$

Notation: the universe of an interpretation I is denoted by $|I|$; for any signature σ and any set U, σ^U stands for the extension of σ obtained by adding distinct new symbols ξ^*, called *names*, for all $\xi \in U$ as object constants. We will identify an interpretation I of σ with its extension to $\sigma^{|I|}$ defined by $I(\xi^*) = \xi$. By σ_f we denote the part of σ consisting of its object and function constants.

An *HT-interpretation* of σ is a triple $\mathcal{I} = \langle \mathcal{I}^\mathrm{f}, \mathcal{I}^\mathrm{h}, \mathcal{I}^\mathrm{t} \rangle$, where

- $\mathcal{I}^\mathrm{f}$ is an interpretation of σ_f, and
- $\mathcal{I}^\mathrm{h}$, $\mathcal{I}^\mathrm{t}$ are sets of atomic formulas formed using predicate constants from σ and the names of elements of $|\mathcal{I}^\mathrm{f}|$ such that $\mathcal{I}^\mathrm{h} \subseteq \mathcal{I}^\mathrm{t}$.

The symbols h ("here") and t ("there") are called *worlds*; they are ordered by the relation h < t. The value that $\mathcal{I}^\mathrm{f}$ assigns to a ground term t of signature $\sigma_\mathrm{f}{}^{|\mathcal{I}^\mathrm{f}|}$ will be denoted by $t^\mathcal{I}$.

The *satisfaction* relation $\models_\mathrm{ht}$ between an HT-interpretation $\mathcal{I}$, a world w, and a first-order sentence F of the signature $\sigma^{|\mathcal{I}^\mathrm{f}|}$, is defined recursively:

- $\mathcal{I}, w \models_\mathrm{ht} p(t_1, \dots, t_k)$ if $p((t_1^\mathcal{I})^*, \dots, (t_k^\mathcal{I})^*) \in \mathcal{I}^w$;
- $\mathcal{I}, w \models_\mathrm{ht} t_1 = t_2$ if $t_1^\mathcal{I} = t_2^\mathcal{I}$;
- $\mathcal{I}, w \not\models_\mathrm{ht} \bot$;
- $\mathcal{I}, w \models_\mathrm{ht} F \wedge G$ if $\mathcal{I}, w \models_\mathrm{ht} F$ and $\mathcal{I}, w \models_\mathrm{ht} G$;
- $\mathcal{I}, w \models_\mathrm{ht} F \vee G$ if $\mathcal{I}, w \models_\mathrm{ht} F$ or $\mathcal{I}, w \models_\mathrm{ht} G$;
- $\mathcal{I}, w \models_\mathrm{ht} F \to G$ if, for every world w' such that $w \leqslant w'$,
$$\mathcal{I}, w' \not\models_\mathrm{ht} F \quad \text{or} \quad \mathcal{I}, w' \models_\mathrm{ht} G;$$
- $\mathcal{I}, w \models_\mathrm{ht} \forall x F(x)$ if, for each $\xi \in |\mathcal{I}^\mathrm{f}|$, $\mathcal{I}, w \models_\mathrm{ht} F(\xi^*)$;
- $\mathcal{I}, w \models_\mathrm{ht} \exists x F(x)$ if, for some $\xi \in |\mathcal{I}^\mathrm{f}|$, $\mathcal{I}, w \models_\mathrm{ht} F(\xi^*)$.

We say that $\mathcal{I}$ *satisfies* F, and write $\mathcal{I} \models_\mathrm{ht} F$, if $\mathcal{I}, \mathrm{h} \models_\mathrm{ht} F$. It is easy to check by induction on F that this condition implies $\mathcal{I}, \mathrm{t} \models_\mathrm{ht} F$.

As shown in [16], system $\mathbf{SQHT}^=$ is sound and complete relative to this semantics: for any set Γ of sentences, a sentence F is derivable from Γ in $\mathbf{SQHT}^=$ iff F is satisfied by every HT-interpretation that satisfies all formulas from Γ.

An interpretation I (in the sense of classical logic) of a signature σ can be represented as a pair $\langle J, X \rangle$, where J is the restriction of I to σ_f, and X is the set of the atomic formulas, formed using predicate constants from σ and the names of elements of $|I|$, which are satisfied by I. The lemma below uses this notation to describe the relationship between the satisfiability relation for HT-interpretations and the transformation $F \mapsto F^*(\mathbf{u})$ introduced in Section 2.3. We assume that σ contains finitely many predicate constants, and the list of these constants is denoted by $\mathbf{p}$. By σ^+ we denote the signature obtained from σ by adding new predicate constants $\mathbf{q}$, one per each member of $\mathbf{p}$. About an atomic formula formed using a predicate constant from σ^+ and names of elements of $|I|$ we say that it is a $\mathbf{p}$-*atom* if its predicate constant belongs to $\mathbf{p}$, and that it is a $\mathbf{q}$-*atom* otherwise. As in Appendix A.1, for any set X of $\mathbf{p}$-atoms we denote by $X^\mathbf{p}_\mathbf{q}$ the set of the $\mathbf{q}$-atoms that are obtained from the elements of X by replacing their predicate constants by the corresponding predicate constants from $\mathbf{q}$.

Lemma 8. *For any HT-interpretation $\mathcal{I}$ and any first-order sentence F of the signature $\sigma^{|\mathcal{I}^\mathrm{f}|}$,*

(i) $\mathcal{I}, \mathrm{t} \models_\mathrm{ht} F$ *iff* $\langle \mathcal{I}^\mathrm{f}, \mathcal{I}^\mathrm{t} \rangle \models F$ *iff* $\langle \mathcal{I}^\mathrm{f}, (\mathcal{I}^\mathrm{h})^\mathbf{p}_\mathbf{q} \cup \mathcal{I}^\mathrm{t} \rangle \models F$;
(ii) $\mathcal{I}, \mathrm{h} \models_\mathrm{ht} F$ *iff* $\langle \mathcal{I}^\mathrm{f}, (\mathcal{I}^\mathrm{h})^\mathbf{p}_\mathbf{q} \cup \mathcal{I}^\mathrm{t} \rangle \models F^*(\mathbf{q})$.

Proof. Each part is easy to check by induction on the size of F. Consider, for instance, the proof of (ii) for the case of implication. We will write I for $\langle \mathcal{I}^\mathrm{f}, (\mathcal{I}^\mathrm{h})^\mathbf{p}_\mathbf{q} \cup \mathcal{I}^\mathrm{t} \rangle$. By the induction hypothesis,

$$\mathcal{I}, \mathrm{h} \models_\mathrm{ht} F \quad \text{iff} \quad I \models F^*(\mathbf{q}),$$
$$\mathcal{I}, \mathrm{h} \models_\mathrm{ht} G \quad \text{iff} \quad I \models G^*(\mathbf{q}).$$

By part (i) of the lemma,

$$\mathcal{I}, \mathrm{t} \models_\mathrm{ht} F \quad \text{iff} \quad I \models F,$$
$$\mathcal{I}, \mathrm{t} \models_\mathrm{ht} G \quad \text{iff} \quad I \models G.$$

Consequently

$$\begin{aligned}
&\mathcal{I}, \mathrm{h} \models_{\mathrm{ht}} F \to G \\
&\quad \text{iff} \quad \big[\mathcal{I}, \mathrm{h} \not\models_{\mathrm{ht}} F \text{ or } \mathcal{I}, \mathrm{h} \models_{\mathrm{ht}} G\big] \quad \text{and} \quad \big[\mathcal{I}, \mathrm{t} \not\models_{\mathrm{ht}} F \text{ or } \mathcal{I}, \mathrm{t} \models_{\mathrm{ht}} G\big] \\
&\quad \text{iff} \quad \big[I \not\models F^*(\mathbf{q}) \text{ or } I \models G^*(\mathbf{q})\big] \quad \text{and} \quad [I \not\models F \text{ or } I \models G] \\
&\quad \text{iff} \quad I \models F^*(\mathbf{q}) \to G^*(\mathbf{q}) \quad \text{and} \quad I \models F \to G \\
&\quad \text{iff} \quad I \models \big(F^*(\mathbf{q}) \to G^*(\mathbf{q})\big) \wedge (F \to G) \\
&\quad \text{iff} \quad I \models (F \to G)^*(\mathbf{q}). \qquad \Box
\end{aligned}$$

A.5.2. First-order equilibrium logic and stable models

An HT-interpretation $\langle \mathcal{I}^{\mathrm{f}}, \mathcal{I}^{\mathrm{h}}, \mathcal{I}^{\mathrm{t}} \rangle$ is *total* if $\mathcal{I}^{\mathrm{h}} = \mathcal{I}^{\mathrm{t}}$. A total HT-interpretation $\langle I, X, X \rangle$ is an *equilibrium model* of a sentence F of the signature $\sigma^{|I|}$ if

(i) $\langle I, X, X \rangle \models_{\mathrm{ht}} F$, and
(ii) for any proper subset Y of X, $\langle I, Y, X \rangle \not\models_{\mathrm{ht}} F$.

It is easy to check by induction on F that condition (i) above is equivalent to $\langle I, X \rangle \models F$.

In the following lemma, σ is a signature containing finitely many predicate constants.

Lemma 9. *For any total HT-interpretation $\langle I, X, X \rangle$ of σ and any first-order sentence F of $\sigma^{|I|}$, $\langle I, X, X \rangle$ is an equilibrium model of F iff $\langle I, X \rangle$ is a* **p***-stable model of F, where* **p** *is the list of all predicate constants of σ.*

Proof. From Lemma 8(ii) we conclude that condition (ii) from the definition of an equilibrium model can be reformulated as follows: for any proper subset Y of X,

$$\big\langle I, Y^{\mathbf{p}}_{\mathbf{q}} \cup X \big\rangle \not\models F^*(\mathbf{q}).$$

This is equivalent to saying that there is no set Y of **p**-atoms such that

$$\big\langle I, Y^{\mathbf{p}}_{\mathbf{q}} \cup X \big\rangle \models (\mathbf{q} < \mathbf{p}) \wedge F^*(\mathbf{q}),$$

and consequently equivalent to the condition

$$\langle I, X \rangle \models \neg\exists\mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F^*(\mathbf{u})\big).$$

It follows that $\langle I, X, X \rangle$ is an equilibrium model of F iff

$$\langle I, X \rangle \models F \wedge \neg\exists\mathbf{u}\big((\mathbf{u} < \mathbf{p}) \wedge F^*(\mathbf{u})\big). \qquad \Box$$

A.5.3. Proof of Theorems 7 and 8

The assertions of Theorems 7 and 8 (Section 5.2) can be jointly reformulated as follows:

For any first-order formulas F and G, the following conditions are equivalent:

(i) *F is strongly equivalent to G,*
(ii) *for any formula H such that every object, function or predicate constant occurring in H occurs in F or in G, and for any occurrence of F in H,* $\mathrm{SM}_{\mathbf{p}^{FG}}[H]$ *is equivalent to* $\mathrm{SM}_{\mathbf{p}^{FG}}[H']$, *where H' is obtained from H by replacing the occurrence of F by G,*
(iii) *formula $F \leftrightarrow G$ is provable in* $\mathbf{SQHT}^{=}$.

The proof repeats, with minor modifications, the argument from [16].

From (i) to (ii): Obvious.

From (ii) to (iii): By $\mathbf{x}$ we will denote the list of variables that are free in F or in G, and we will write F as $F(\mathbf{x})$, and G as $G(\mathbf{x})$. Our goal is to show that $F(\mathbf{x}) \leftrightarrow G(\mathbf{x})$ is provable in $\mathbf{SQHT}^{=}$. Without loss of generality, we can assume that every predicate constant in the underlying signature σ belongs to $\mathbf{p}^{FG}$. Take an HT-interpretation $\mathcal{I}$ and a tuple $\mathbf{c}$ of names of the same length as $\mathbf{x}$. We need to show that $\mathcal{I}$ satisfies $F(\mathbf{c})$ iff $\mathcal{I}$ satisfies $G(\mathbf{c})$. Assume, for instance, that $\mathcal{I} \models_{\mathrm{ht}} F(\mathbf{c})$, and denote the formula *Choice*$(\mathbf{p}^{FG})$ by C. *Case 1*: $\mathcal{I}$ is total. By (ii),

$$\mathrm{SM}_{\mathbf{p}^{FG}}\big[F(\mathbf{x}) \wedge C\big] \text{ is equivalent to } \mathrm{SM}_{\mathbf{p}^{FG}}\big[G(\mathbf{x}) \wedge C\big],$$

and consequently

$$\mathrm{SM}_{\mathbf{p}^{FG}}\big[F(\mathbf{c}) \wedge C\big] \text{ is equivalent to } \mathrm{SM}_{\mathbf{p}^{FG}}\big[G(\mathbf{c}) \wedge C\big].$$

By Lemma 9, it follows that the sentences

$$F(\mathbf{c}) \wedge C, \quad G(\mathbf{c}) \wedge C \tag{A.4}$$

have the same equilibrium models. Since $\mathcal{I}$ is total and satisfies $F(\mathbf{c})$, $\mathcal{I}$ is an equilibrium model of the first of the formulas (A.4). Consequently, it is an equilibrium model of the second, so that $\mathcal{I} \models_{\text{ht}} G(\mathbf{c})$. *Case 2*: $\mathcal{I}$ is not total. Let $\mathcal{J}$ be the total HT-interpretation $\langle \mathcal{I}^{\text{f}}, \mathcal{I}^{\text{t}}, \mathcal{I}^{\text{t}} \rangle$. From the assumption $\mathcal{I} \models_{\text{ht}} F(\mathbf{c})$ we can conclude that $\mathcal{I}, \text{t} \models_{\text{ht}} F(\mathbf{c})$, and, by Lemma 8(i), that $\mathcal{J} \models_{\text{ht}} F(\mathbf{c})$. Furthermore, by reasoning as in Case 1 with $\mathcal{J}$ in place of $\mathcal{I}$, we conclude that $\mathcal{J} \models_{\text{ht}} G(\mathbf{c})$. By (ii),

$$\text{SM}_{\mathbf{p}^{FG}}\big[F(\mathbf{x}) \wedge \big(G(\mathbf{x}) \to C\big)\big]$$

is equivalent to

$$\text{SM}_{\mathbf{p}^{FG}}\big[G(\mathbf{x}) \wedge \big(G(\mathbf{x}) \to C\big)\big],$$

and consequently

$$\text{SM}_{\mathbf{p}^{FG}}\big[F(\mathbf{c}) \wedge \big(G(\mathbf{c}) \to C\big)\big]$$

is equivalent to

$$\text{SM}_{\mathbf{p}^{FG}}\big[G(\mathbf{c}) \wedge \big(G(\mathbf{c}) \to C\big)\big].$$

By Lemma 9, it follows that the sentences

$$F(\mathbf{c}) \wedge \big(G(\mathbf{c}) \to C\big), \quad G(\mathbf{c}) \wedge \big(G(\mathbf{c}) \to C\big) \tag{A.5}$$

have the same equilibrium models. The latter can be rewritten as

$$G(\mathbf{c}) \wedge C. \tag{A.6}$$

Since $\mathcal{J}$ is a total HT-interpretation satisfying $G(\mathbf{c})$, it is an equilibrium model of (A.6). Consequently, $\mathcal{J}$ is an equilibrium model of the first of the formulas (A.5). Hence that formula is not satisfied by $\mathcal{I}$. Since its first conjunctive term $F(\mathbf{c})$ is satisfied by $\mathcal{I}$, we conclude that $\mathcal{I}$ does not satisfy the second term $G(\mathbf{c}) \to C$. Since $\mathcal{I}, \text{t} \models_{\text{ht}} C$, this is only possible when $\mathcal{I}, \text{h} \models_{\text{ht}} G(\mathbf{c})$, that is, $\mathcal{I} \models_{\text{ht}} G(\mathbf{c})$.

From (iii) to (i): Let H' be obtained from H by replacing an occurrence of F by G, and let $\mathbf{p}$ be a list of predicate constants. We will denote by $\mathbf{x}$ the list of variables that are free in at least one of the formulas H, H', and we will write H as $H(\mathbf{x})$, and H' as $H'(\mathbf{x})$. Our goal is to show that $\text{SM}_{\mathbf{p}}[H(\mathbf{x})]$ is equivalent to $\text{SM}_{\mathbf{p}}[H'(\mathbf{x})]$. Without loss of generality we can assume that every predicate constant in the underlying signature σ occurs in $H(\mathbf{x})$ or $H'(\mathbf{x})$, so that the set of predicate constants in σ is finite. Let $\mathbf{q}$ be the list of predicate constants from σ that do not belong to $\mathbf{p}$. By Theorem 2, it is sufficient to prove that $\text{SM}_{\mathbf{pq}}[H'(\mathbf{x}) \wedge \mathit{Choice}(\mathbf{q})]$ is equivalent to $\text{SM}_{\mathbf{pq}}[H(\mathbf{x}) \wedge \mathit{Choice}(\mathbf{q})]$. Take an interpretation $\langle I, X \rangle$ of σ and a tuple $\mathbf{c}$ of names, of the same length as $\mathbf{x}$. We need to show that $H'(\mathbf{c}) \wedge \mathit{Choice}(\mathbf{q})$ and $H(\mathbf{c}) \wedge \mathit{Choice}(\mathbf{q})$ have the same $\mathbf{pq}$-stable models. By Lemma 9, this is equivalent to saying that these two sentences have the same equilibrium models. It remains to note that the equivalence between these two sentences is provable in $\mathbf{SQHT}^=$, and consequently these sentences are satisfied by the same HT-interpretations. □

A.5.4. Proof of Theorem 5

Theorem 5. *For any first-order formulas F and G, if the formula $F \leftrightarrow G$ is derivable in* $\mathbf{SQHT}^=$ *from the formulas $\mathit{Choice}(q)$ for the extensional predicates q then* SM$[F]$ *is equivalent to* SM$[G]$.

Proof. Let $\mathbf{p}$ be the list of intensional predicates, and let $\mathbf{q}$ be the list of all other predicate constants occurring in F or in G. Since $F \leftrightarrow G$ is derivable in $\mathbf{SQHT}^=$ from $\mathit{Choice}(\mathbf{q})$, the formula

$$F \wedge \mathit{Choice}(\mathbf{q}) \leftrightarrow G \wedge \mathit{Choice}(\mathbf{q})$$

is provable in $\mathbf{SQHT}^=$. By Theorem 8, it follows that the left-hand side is strongly equivalent to the right-hand side. It follows that $\text{SM}_{\mathbf{pq}}[F \wedge \mathit{Choice}(\mathbf{q})]$ is equivalent to $\text{SM}_{\mathbf{pq}}[G \wedge \mathit{Choice}(\mathbf{q})]$. By Theorem 2, we can conclude that $\text{SM}_{\mathbf{p}}[F]$ is equivalent to $\text{SM}_{\mathbf{p}}[G]$. □

A.5.5. Proof of Theorem 6

Lemma 10. *If a formula G has no strictly positive occurrences of predicate constants from a list* $\mathbf{p}$ *then $G \leftrightarrow \neg\neg G$ is derivable in* $\mathbf{SQHT}^=$ *from the formulas $\mathit{Choice}(q)$ for the predicate constants q that occur in G but do not belong to* $\mathbf{p}$.

Proof. By induction on G, using the fact that the equivalences

$$\neg\neg(F \wedge G) \leftrightarrow \neg\neg F \wedge \neg\neg G,$$

$$\neg\neg(F \vee G) \leftrightarrow \neg\neg F \vee \neg\neg G,$$

$$\neg\neg(F \to G) \leftrightarrow F \to \neg\neg G$$

are provable in $\mathbf{SQHT}^=$. □

Theorem 6. *Let F' be the formula obtained from a first-order formula F by inserting $\neg\neg$ in front of a subformula G. If G has no strictly positive occurrences of intensional predicates then* SM$[F']$ *is equivalent to* SM$[F]$.

Proof. Immediate from Lemma 10 and Theorem 5. □

A.6. Proof of Theorem 9

Theorem 9. *F is strongly equivalent to G iff the formula*

$$\big(\mathbf{q} \leqslant \mathbf{p}^{FG}\big) \to \big(F^*(\mathbf{q}) \leftrightarrow G^*(\mathbf{q})\big) \tag{A.7}$$

is logically valid.

Proof. Without loss of generality, we can assume that every predicate constant in the underlying signature σ belongs to $\mathbf{p}^{FG}$. By $\mathbf{x}$ we will denote the list of variables that are free in F or in G, and we will write F as $F(\mathbf{x})$, G as $G(\mathbf{x})$, $F^*(\mathbf{q})$ as $F^*(\mathbf{q},\mathbf{x})$, $G^*(\mathbf{q})$ as $G^*(\mathbf{q},\mathbf{x})$, and $\mathbf{p}^{FG}$ as $\mathbf{p}$.

By Theorem 8, the condition

$F(\mathbf{x})$ is strongly equivalent to $G(\mathbf{x})$

is equivalent to the condition

$F(\mathbf{x}) \leftrightarrow G(\mathbf{x})$ is provable in $\mathbf{SQHT}^=$.

It can be further reformulated as follows:

for any HT-interpretation $\langle I, Y, X\rangle$

and for any tuple $\mathbf{c}$ of names of the same length as $\mathbf{x}$,

$\langle I, Y, X\rangle \models_{\text{ht}} F(\mathbf{c})$ iff $\langle I, Y, X\rangle \models_{\text{ht}} G(\mathbf{c})$.

By Lemma 8(ii), the last line can be equivalently rewritten as

$\big\langle I, Y^{\mathbf{p}}_{\mathbf{q}} \cup X\big\rangle \models F^*(\mathbf{q},\mathbf{c})$ iff $\big\langle I, Y^{\mathbf{p}}_{\mathbf{q}} \cup X\big\rangle \models G^*(\mathbf{q},\mathbf{c})$.

Consequently $F(\mathbf{x})$ is strongly equivalent to $G(\mathbf{x})$ iff

for any interpretation I of σ_f, any sets X and Y of $\mathbf{p}$-atoms,

and any tuple $\mathbf{c}$ of names of the same length as $\mathbf{x}$,

$\big\langle I, Y^{\mathbf{p}}_{\mathbf{q}} \cup X\big\rangle \models \mathbf{q} \leqslant \mathbf{p} \wedge F^*(\mathbf{q},\mathbf{c})$ iff $\big\langle I, Y^{\mathbf{p}}_{\mathbf{q}} \cup X\big\rangle \models \mathbf{q} \leqslant \mathbf{p} \wedge G^*(\mathbf{q},\mathbf{c})$.

This condition is equivalent to the logical validity of (A.7). □

A.7. Proof of Theorems 10 and 12

Theorem 10 follows from part (a) of Theorem 12, so that we only need to prove the latter. Let the intensional predicates be $p_1,\dots,p_n$. By $\mathbf{e}^i(\mathbf{x}^i)$ we denote the tuple

$$p_1,\dots,p_{i-1},\lambda\mathbf{y}^i\big(p_i\big(\mathbf{y}^i\big) \wedge \mathbf{y}^i \neq \mathbf{x}^i\big), p_{i+1},\dots,p_n,$$

where $\mathbf{y}^i$ is a tuple of new distinct variables.

Lemma 11. *For any formula F, the implications*

$$F^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big) \to F \quad (i = 1,\dots,n)$$

are logically valid.

Proof. Immediate from Lemma 5. □

Lemma 12. *If a formula F does not contain strictly positive occurrences of p_i then $F^*(\mathbf{e}^i(\mathbf{x}^i))$ is equivalent to F.*

Proof. Immediate from Lemma 7 with p_i as $\mathbf{p}$. $\square$

Recall that a formula in Clark normal form can be written as

$$\bigwedge_{i=1}^{n} \forall \mathbf{x}^i \big(G_i \to p_i\big(\mathbf{x}^i\big)\big), \tag{A.8}$$

where each $\mathbf{x}^i$ is a list of distinct variables.

Lemma 13. *If F is* (A.8) *then* PSM$[F]$ *is equivalent to*

$$F \wedge \bigwedge_{i=1}^{n} \forall \mathbf{x}^i \big(p_i\big(\mathbf{x}^i\big) \to G_i^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big)\big).$$

Proof. As discussed in Section 7.2, PSM$[F]$ is equivalent to the first-order formula

$$F \wedge \neg\big(F^*\big)^{(1)}_{\mathbf{u}}(\mathbf{p}).$$

Formula $(F^*)^{(1)}_{\mathbf{u}}(\mathbf{p})$ can be written as

$$\bigvee_i \big(\exists \mathbf{x}^i \big(p_i\big(\mathbf{x}^i\big) \wedge F^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big)\big)\big).$$

Consequently, PSM$[F]$ can be equivalently rewritten as

$$F \wedge \bigwedge_i \forall \mathbf{x}^i \big(p_i\big(\mathbf{x}^i\big) \to \neg F^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big)\big).$$

To prove the assertion of the lemma, it remains to derive the equivalence between

$$\neg F^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big) \tag{A.9}$$

and

$$G_i^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big) \tag{A.10}$$

from assumption F.

Formula $F^*(\mathbf{u})$ can be rewritten, under assumption F, as the conjunction of the formulas

$$\forall \mathbf{y}^j \big(G_j^*(\mathbf{u}) \to u_j\big(\mathbf{y}^j\big)\big)$$

for all $j = 1, \dots, n$. The j-th term of the tuple $\mathbf{e}^i(\mathbf{x}^i)$ is $\lambda \mathbf{y}^i(p_i(\mathbf{y}^i) \wedge \mathbf{y}^i \neq \mathbf{x}^i)$ if $j = i$, and p_j otherwise. Consequently, the j-th conjunctive term of $F^*(\mathbf{e}^i(\mathbf{x}^i))$ is

$$\forall \mathbf{y}^i \big(G_i^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big) \to \big(p_i\big(\mathbf{y}^i\big) \wedge \mathbf{y}^i \neq \mathbf{x}^i\big)\big), \tag{A.11}$$

if $j = i$, and

$$\forall \mathbf{y}^j \big(G_j^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big) \to p_j\big(\mathbf{y}^j\big)\big) \tag{A.12}$$

otherwise. Lemma 11 shows that in the presence of the conjunctive term

$$\forall \mathbf{y}^j \big(G_j \to p_j\big(\mathbf{y}^j\big)\big)$$

of F, the conjunctive term (A.11) of $F^*(\mathbf{e}^i(\mathbf{x}^i))$ can be rewritten as

$$\forall \mathbf{y}^i \big(G_i^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big) \to \mathbf{y}^i \neq \mathbf{x}^i\big), \tag{A.13}$$

and the other conjunctive terms (A.12) can be dropped altogether. We conclude that formula (A.9) can be written as

$$\neg \forall \mathbf{y}^i \big(G_i^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big) \to \mathbf{y}^i \neq \mathbf{x}^i\big),$$

which is equivalent to

$$\exists \mathbf{y}^i \big(G_i^*\big(\mathbf{e}^i\big(\mathbf{x}^i\big)\big) \wedge \mathbf{y}^i = \mathbf{x}^i\big), \tag{A.14}$$

and consequently to (A.10). $\square$

Theorem 12. *For any formula F in Clark normal form,* (a) *the implication*

$$\mathrm{PSM}[F] \rightarrow \mathrm{Comp}[F]$$

is logically valid; (b) *if F is pure then* $\mathrm{PSM}[F]$ *is equivalent to* $\mathrm{Comp}[F]$.

Proof. If F is (A.8) then $\mathrm{Comp}[F]$ is equivalent to

$$F \wedge \bigwedge_i \forall \mathbf{x}^i \big(p_i(\mathbf{x}^i) \rightarrow G_i\big). \tag{A.15}$$

On the other hand, by Lemma 13, $\mathrm{PSM}[F]$ is equivalent to

$$F \wedge \bigwedge_i \forall \mathbf{x}^i \big(p_i(\mathbf{x}^i) \rightarrow G_i^*(\mathbf{e}^i(\mathbf{x}^i))\big).$$

Claim (a) follows by Lemma 11. To prove claim (b), note that when F is pure then $G_i^*(\mathbf{e}^i(\mathbf{x}^i))$ is equivalent to G_i by Lemma 12. □

A.8. Proofs of Theorems 11 and 13

Since every tight program is pure, Theorem 11 follows from Theorem 12(b) and Theorem 13. Consequently we only need to prove Theorem 13.

In the following lemma, F is a first-order formula, $\mathbf{p}$ is the list of intensional predicates $p_1, \dots, p_n$, and $\mathbf{u}$ is a tuple of distinct predicate variables $u_1, \dots, u_n$.

Lemma 14. *Let S be the set of i's such that p_i has a strictly positive occurrence in F. The formula*

$$\Big((\mathbf{u} \leqslant \mathbf{p}) \wedge \bigwedge_{i \in S} (u_i = p_i)\Big) \rightarrow \big(F \leftrightarrow F^*(\mathbf{u})\big)$$

is logically valid.

Proof. By induction on F. We will consider the case when F is $G \rightarrow H$; the other cases are straightforward. It is sufficient to derive the implication

$$(G \rightarrow H) \rightarrow \big(G^*(\mathbf{u}) \rightarrow H^*(\mathbf{u})\big) \tag{A.16}$$

from the assumption

$$(\mathbf{u} \leqslant \mathbf{p}) \wedge \bigwedge_{i \in S} (u_i = p_i). \tag{A.17}$$

Since every i such that p_i has a strictly positive occurrence in H belongs to S, it follows from the induction hypothesis that the implication

$$\Big((\mathbf{u} \leqslant \mathbf{p}) \wedge \bigwedge_{i \in S} (u_i = p_i)\Big) \rightarrow \big(H \leftrightarrow H^*(\mathbf{u})\big) \tag{A.18}$$

is logically valid. By Lemma 5, the implication

$$(\mathbf{u} \leqslant \mathbf{p}) \wedge G^*(\mathbf{u}) \rightarrow G \tag{A.19}$$

is logically valid also. It remains to observe that (A.16) is a propositional consequence of (A.17), (A.18), and (A.19). □

Recall that an occurrence of a predicate constant in a formula is called positive if the number of implications containing that occurrence in the antecedent is even (Section 4.3); if that number is odd then the occurrence is *negative*. Negative occurrences should be distinguished from negated occurrences—those belonging to a subformula of the form $F \rightarrow \bot$ (Section 6.2). In the following lemmas, $\mathbf{v}$ is a tuple of distinct predicate variables disjoint from $\mathbf{u}$.

Lemma 15. *Let S^+ be the set of i's such that p_i has a positive nonnegated occurrence in F, and let S^- be the set of i's such that p_i has a negative nonnegated occurrence in F. The formulas*

(a) $((\mathbf{u} \leqslant \mathbf{v}) \wedge (\mathbf{v} \leqslant \mathbf{p}) \wedge \bigwedge_{i \in S^+} (u_i = p_i)) \rightarrow (F^*(\mathbf{v}) \rightarrow F^*(\mathbf{u}))$,
(b) $((\mathbf{u} \leqslant \mathbf{v}) \wedge (\mathbf{v} \leqslant \mathbf{p}) \wedge \bigwedge_{i \in S^-} (u_i = p_i)) \rightarrow (F^*(\mathbf{u}) \rightarrow F^*(\mathbf{v}))$

are logically valid.

Proof. Both parts are proved simultaneously by induction on F. We will only consider the proof of (a) in the case when F is an implication $G \to H$. *Case 1:* H is $\bot$, so that F is $\neg G$. By Lemma 6, the formulas

$$\mathbf{u} \leqslant \mathbf{p} \to \big(F^*(\mathbf{u}) \leftrightarrow F\big),$$
$$\mathbf{v} \leqslant \mathbf{p} \to \big(F^*(\mathbf{v}) \leftrightarrow F\big)$$

are logically valid. Consequently formula (a) is logically valid also. *Case 2:* H is different from $\bot$. Then each p_i that has a nonnegated occurrence in G or H has a nonnegated occurrence in F as well. Denote the antecedent of (a) by *Ant*; then (a) can be written as

$$Ant \to \Big(\big(F \wedge \big(G^*(\mathbf{v}) \to H^*(\mathbf{v})\big)\big) \to \big(F \wedge \big(G^*(\mathbf{u}) \to H^*(\mathbf{u})\big)\big)\Big). \tag{A.20}$$

By part (b) of the induction hypothesis applied to G, the formula

$$Ant \to \big(G^*(\mathbf{u}) \to G^*(\mathbf{v})\big) \tag{A.21}$$

is logically valid. By part (a) of the induction hypothesis applied to H, the formula

$$Ant \to \big(H^*(\mathbf{v}) \to H^*(\mathbf{u})\big) \tag{A.22}$$

is logically valid. It remains to observe that (A.20) is a propositional consequence of (A.21) and (A.22). □

Lemma 16. *Let D be the set of edges of the predicate dependency graph of F. The formula*

$$\Bigg((\mathbf{u} \leqslant \mathbf{v}) \wedge (\mathbf{v} \leqslant \mathbf{p}) \wedge \bigwedge_{i,j:\ (p_i,p_j)\in D} (u_j = p_j \vee v_i = p_i)\Bigg) \to \big(F^*(\mathbf{u}) \to F^*(\mathbf{v})\big)$$

is logically valid.

Proof. By induction on F. We will only consider the case when F is an implication $G \to H$. Let *Ant* be the antecedent

$$(\mathbf{u} \leqslant \mathbf{v}) \wedge (\mathbf{v} \leqslant \mathbf{p}) \wedge \bigwedge_{i,j:\ (p_i,p_j)\in D} (u_j = p_j \vee v_i = p_i)$$

of the formula in question, and let S be the set of i's such that p_i has a strictly positive occurrence in F. It is sufficient to establish the logical validity of the formulas

$$\Bigg(Ant \wedge \bigwedge_{i\in S} v_i = p_i\Bigg) \to \big(F^*(\mathbf{u}) \to F^*(\mathbf{v})\big) \tag{A.23}$$

and

$$(Ant \wedge v_i \neq p_i) \to \big(F^*(\mathbf{u}) \to F^*(\mathbf{v})\big) \quad (i \in S). \tag{A.24}$$

From Lemma 14 we conclude that the formula

$$\Bigg(Ant \wedge \bigwedge_{i\in S} v_i = p_i\Bigg) \to \big(F \leftrightarrow F^*(\mathbf{v})\big)$$

is logically valid; (A.23) is a propositional consequence of this formula, in view of the fact that F is a conjunctive term of $F^*(\mathbf{u})$. Formula (A.24) is a propositional consequence of

$$(Ant \wedge v_i \neq p_i) \to \Big(\big(G^*(\mathbf{u}) \to H^*(\mathbf{u})\big) \to \big(G^*(\mathbf{v}) \to H^*(\mathbf{v})\big)\Big), \tag{A.25}$$

so that the proof will be completed if we establish the logical validity of the latter for each $i \in S$.

Note first that every edge of the dependency graph of H is an edge of the dependency graph of F. Consequently the induction hypothesis implies that the formula

$$Ant \to \big(H^*(\mathbf{u}) \to H^*(\mathbf{v})\big) \tag{A.26}$$

is logically valid. Furthermore, it is clear from the definition of *Ant* that the formula

$$(Ant \wedge v_i \neq p_i) \to \bigwedge_{j:\ (p_i,p_j)\in D} u_j = p_j$$

is a tautology. Let S^+ be the set of j's such that p_j has a positive nonnegated occurrence in G. By the definition of the predicate dependency graph, $(p_i, p_j) \in D$ whenever $i \in S$ and $j \in S^+$. Consequently

$$(Ant \wedge v_i \neq p_i) \to \bigwedge_{j\in S^+} u_j = p_j$$

is a tautology also. In view of Lemma 15(a), it follows that the formula

$$(Ant \wedge v_i \neq p_i) \to \big(G^*(\mathbf{v}) \to G^*(\mathbf{u})\big) \tag{A.27}$$

is logically valid. It remains to observe that (A.25) is a propositional consequence of (A.26) and (A.27). □

Theorem 13. *For any tight formula F,* PSM$[F]$ *is equivalent to* SM$[F]$.

Proof. We only need to prove the implication left-to-right. Since F is tight, we can assume without loss of generality that the members $p_1, \dots, p_n$ of $\mathbf{p}$ are ordered in such a way that $i < j$ for all edges (p_i, p_j) of the dependency graph of F. Assume PSM$[F]$ and $\mathbf{u} < \mathbf{p}$; we need to derive $\neg F^*(\mathbf{u})$. Let m be the largest i such that $u_i \neq p_i$. Take $\mathbf{x}$ such that $p_m(\mathbf{x}) \wedge \neg u_m(\mathbf{x})$. Choose $\mathbf{v}$ as follows: v_i is $\lambda\mathbf{y}(p_i(\mathbf{y}) \wedge \mathbf{x} \neq \mathbf{y})$ if $i = m$, and p_i otherwise. Then

$$(\mathbf{u} \leqslant \mathbf{v}) \wedge (\mathbf{v} \leqslant \mathbf{p}) \wedge \bigwedge_{i,j:\ (p_i,p_j)\in D} (u_j = p_j \vee v_i = p_i). \tag{A.28}$$

Indeed, the conjunctive terms $\mathbf{u} \leqslant \mathbf{v}$ and $\mathbf{v} \leqslant \mathbf{p}$ are immediate, as well as the second disjunctive term of $u_j = p_j \vee v_i = p_i$ for any i different from m. Any j such that $(p_m, p_j) \in D$ is greater than m; by the choice of m, we get the first disjunctive term $u_j = p_j$. From (A.28) and the formula from Lemma 16,

$$F^*(\mathbf{u}) \to F^*(\mathbf{v}).$$

On the other hand, $\mathbf{v} \overset{1}{<} \mathbf{p}$, so that, in view of PSM$[F]$, we can conclude that $\neg F^*(\mathbf{v})$. Consequently $\neg F^*(\mathbf{u})$. □

References

[1] Keith Clark, Negation as failure, in: Herve Gallaire, Jack Minker (Eds.), Logic and Data Bases, Plenum Press, New York, 1978, pp. 293–322.
[2] Esra Erdem, Vladimir Lifschitz, Tight logic programs, Theory and Practice of Logic Programming 3 (2003) 499–518.
[3] Wolfgang Faber, Nicola Leone, Gerald Pfeifer, Recursive aggregates in disjunctive logic programs: Semantics and complexity,[9] in: Proceedings of European Conference on Logics in Artificial Intelligence (JELIA), 2004
[4] François Fages, Consistency of Clark's completion and existence of stable models, Journal of Methods of Logic in Computer Science 1 (1994) 51–60.
[5] Paolo Ferraris, Vladimir Lifschitz, Mathematical foundations of answer set programming, in: We Will Show Them! Essays in Honour of Dov Gabbay, King's College Publications, 2005, pp. 615–664.
[6] Paolo Ferraris, Joohyung Lee, Vladimir Lifschitz, A new perspective on stable models, in: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 2007, pp. 372–379.
[7] Paolo Ferraris, Joohyung Lee, Vladimir Lifschitz, Ravi Palla, Symmetric splitting in the general theory of stable models, in: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 2009, pp. 797–803.
[8] Paolo Ferraris, Answer sets for propositional theories, in: Proceedings of International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR), 2005, pp. 119–131.
[9] Michael Gelfond, Vladimir Lifschitz, The stable model semantics for logic programming, in: Robert Kowalski, Kenneth Bowen (Eds.), Proceedings of International Logic Programming Conference and Symposium, MIT Press, 1988, pp. 1070–1080.
[10] Michael Gelfond, Vladimir Lifschitz, Classical negation in logic programs and disjunctive databases, New Generation Computing 9 (1991) 365–385.
[11] Carla P. Gomes, Henry Kautz, Ashish Sabharwal, Bart Selman, Satisfiability solvers, in: Frank van Harmelen, Vladimir Lifschitz, Bruce Porter (Eds.), Handbook of Knowledge Representation, Elsevier, 2008, pp. 89–134.
[12] Joohyung Lee, Fangzhen Lin, Loop formulas for circumscription, Artificial Intelligence 170 (2) (2006) 160–185.
[13] Joohyung Lee, Yunsong Meng, On loop formulas with variables, in: Proceedings of the International Conference on Knowledge Representation and Reasoning (KR), 2008, pp. 444–453.
[14] Joohyung Lee, Vladimir Lifschitz, Ravi Palla, A reductive semantics for counting and choice in answer set programming, in: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2008, pp. 472–479.
[15] Vladimir Lifschitz, David Pearce, Agustin Valverde, Strongly equivalent logic programs, ACM Transactions on Computational Logic 2 (2001) 526–541.
[16] Vladimir Lifschitz, David Pearce, Agustin Valverde, A characterization of strong equivalence for logic programs with variables, in: Proceedings of International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR), 2007.
[17] Vladimir Lifschitz, Computing circumscription, in: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 1985, pp. 121–127.
[18] Vladimir Lifschitz, Pointwise circumscription, in: Matthew Ginsberg (Ed.), Readings in Nonmonotonic Reasoning, Morgan Kaufmann, San Mateo, CA, 1987, pp. 179–193.
[19] Vladimir Lifschitz, Twelve definitions of a stable model, in: Proceedings of International Conference on Logic Programming (ICLP), 2008, pp. 37–51.
[20] Fangzhen Lin, Raymond Reiter, Rules as actions: A situation calculus semantics for logic programs, Journal of Logic Programming 31 (1997) 299–330.
[21] Fangzhen Lin, Shoham Yoav, A logic of knowledge and justified assumptions, Artificial Intelligence 57 (1992) 271–289.
[22] Fangzhen Lin, Yuting Zhao, ASSAT: Computing answer sets of a logic program by SAT solvers, in: Proceedings of National Conference on Artificial Intelligence (AAAI), MIT Press, 2002, pp. 112–117.
[23] Fangzhen Lin, Yi Zhou, From answer set logic programming to circumscription via logic of GK, in: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 2007.
[24] Fangzhen Lin, A study of nonmonotonic reasoning, PhD thesis, Stanford University, 1991.
[25] Victor Marek, Mirosław Truszczyński, Stable models and an alternative logic programming paradigm, in: The Logic Programming Paradigm: A 25-Year Perspective, Springer-Verlag, 1999, pp. 375–398.
[26] John McCarthy, Circumscription—a form of non-monotonic reasoning, Artificial Intelligence 13 (1980) 27–39, 171–172.
[27] John McCarthy, Applications of circumscription to formalizing common sense knowledge, Artificial Intelligence 26 (3) (1986) 89–116.

[9] Revised version: http://www.wfaber.com/research/papers/jelia2004.pdf.

[28] Robert Moore, Semantical considerations on nonmonotonic logic, Artificial Intelligence 25 (1) (1985) 75–94.
[29] Ilkka Niemelä, Logic programs with stable model semantics as a constraint programming paradigm, Annals of Mathematics and Artificial Intelligence 25 (1999) 241–273.
[30] Emilia Oikarinen, Tomi Janhunen, Achieving compositionality of the stable model semantics for Smodels programs, Theory and Practice of Logic Programming 5–6 (2008) 717–761.
[31] David Pearce, Hans Tompits, Stefan Woltran, Encodings for equilibrium logic and logic programs with nested expressions, in: Proceedings of Portuguese Conference on Artificial Intelligence (EPIA), 2001, pp. 306–320.
[32] David Pearce, A new logical characterization of stable models and answer sets, in: Jürgen Dix, Luis Pereira, Teodor Przymusinski (Eds.), Non-Monotonic Extensions of Logic Programming, in: Lecture Notes in Artificial Intelligence, vol. 1216, Springer-Verlag, 1997, pp. 57–70.
[33] Raymond Reiter, A logic for default reasoning, Artificial Intelligence 13 (1980) 81–132.
[34] Patrik Simons, Ilkka Niemelä, Timo Soininen, Extending and implementing the stable model semantics, Artificial Intelligence 138 (2002) 181–234.
[35] Mark Wallace, Tight, consistent and computable completions for unrestricted logic programs, Journal of Logic Programming 15 (1993) 243–273.