8-22-2022

Positive Dependency Graphs Revisited

Jorge Fandinno
University of Nebraska at Omaha, jfandinno@unomaha.edu

Vladimir Lifschitz
University of Texas at Austin

doi:10.1017/S1471068422000333

*Positive Dependency Graphs Revisited**

JORGE FANDINNO
University of Nebraska Omaha, Omaha, NE 68182, USA
(*e-mail:* jfandinno@unomaha.edu)

VLADIMIR LIFSCHITZ
University of Texas at Austin, Austin, TX 78712, USA
(*e-mail:* lifschitzv@gmail.com)

submitted 12 October 2021; revised 7 April 2022; accepted 20 July 2022

Abstract

Theory of stable models is the mathematical basis of answer set programming. Several results in that theory refer to the concept of the positive dependency graph of a logic program. We describe a modification of that concept and show that the new understanding of positive dependency makes it possible to strengthen some of these results.

KEYWORDS: answer set programming, stable models, completion, splitting, loop formulas

1 Introduction

This note contributes to the theory of stable models, which serves as the mathematical basis of answer set programming (Marek and Truszczynski 1999; Niemelä 1999; Lifschitz 2019). Several results in that theory refer to "positive dependencies" between atoms in a logic program – the idea used by François Fages (Fages 1994) for the purpose of describing the relationship between program completion (Clark 1978) and stable models (Gelfond and Lifschitz 1988). It was applied later to the study of loops and to designing the answer set solver ASSAT (Lin and Zhao 2004) and found other applications.

For a program consisting of rules of the form:

$$H \leftarrow B_1, \dots, B_m, not\ B_{m+1}, \dots, not\ B_n, \tag{1}$$

where $H, B_1, \dots, B_n$ are propositional atoms, and the *positive dependency graph* is defined as the directed graph such that

- its vertices are the atoms occurring in the program, and
- its edges go from H to $B_1, \dots, B_m$ for all rules (1) of the program.

For example, the positive dependency graph of the program:

$$\begin{aligned} q &\leftarrow p, \\ p &\leftarrow q, not\ r \end{aligned} \tag{2}$$

has two edges, (q, p) and (p, q).

* Thanks to Paolo Ferraris, Joohyung Lee, Yuliya Lierler, and the anonymous referees for comments on earlier versions of this note.

In the early days of answer set programming, the syntactic form of every rule of a program was similar to (1) so that the definition of the positive dependency graph aforementioned was applicable to all grounded programs. Later on, the syntax of rules was extended in several ways. In one of these generalizations, reviewed in Section 2, rules are replaced by arbitrary propositional formulas (Ferraris 2005). Rule (1) can be viewed as a special case – as alternative notation for the implication:

$$B_1 \wedge \cdots \wedge B_m \wedge \neg B_{m+1} \wedge \cdots \neg B_n \to H.$$

This degree of generality is important in connection with the use of aggregates, such as the cardinality of a set, in the body of a rule (Ferraris 2005, Section 4).

A generalization of the definition of the positive dependency graph to propositional formulas (Ferraris *et al*. 2006) and further generalizations have been used for several purposes:

- (i) to extend Fages' theorem on tight programs (Fages 1994) to first-order formulas (Ferraris *et al*. 2011) and to infinitary propositional formulas (Lifschitz and Yang 2013),
- (ii) to extend the theory of loops (Lin and Zhao 2004) to arbitrary propositional formulas (Ferraris *et al*. 2006),
- (iii) to investigate a logic programming counterpart of pointwise circumscription (Lifschitz 1986) in the context of first-order formulas (Ferraris *et al*. 2011),
- (iv) to extend the process of symmetric splitting (Oikarinen and Janhunen 2008) to first-order formulas (Ferraris *et al*. 2009) and to infinitary propositional formulas (Harrison and Lifschitz 2016).

In this note, we reexamine the definition of the positive dependency graph used in these publications and argue that a different interpretation of positive dependency would be more appropriate in two of these research lines – in those listed above under (i) and (iii). Two theorems on properties of modified dependency graphs are stated in Sections 4.1 and 4.3 and proved in Section 5. The possibility of extending Fages' theorem along the lines of Theorem 1 is used in the proof of a theorem on the verification of locally tight programs (Fandinno and Lifschitz 2021).

2 Review: Stable models of propositional theories

We assume that formulas are built from propositional atoms and the symbol $\bot$ using the binary connectives $\wedge$, $\vee$, $\to$; $\neg F$ stands for $F \to \bot$, and $F \leftrightarrow G$ stands for $(F \to G) \wedge (G \to F)$. A *propositional theory* is a set of formulas. An *interpretation* is a set of atoms; we identify an interpretation I with the truth assignment that maps the elements of I to *true* and all other atoms to *false*.

The *reduct* F^I of a formula F with respect to an interpretation I is the formula obtained from F by replacing every maximal subformula of F that is not satisfied by I with $\bot$ (Ferraris 2005, Section 2.1). The reduct T^I of a propositional theory T is the set of the reducts F^I of all formulas F in T. An interpretation I is a *stable model* of a propositional theory T if it is minimal (with respect to set inclusion) among the models of T^I.

https://doi.org/10.1017/S1471068422000333 Published online by Cambridge University Press

Consider, for instance, the formulas:

$$\begin{aligned} &p \to q, \\ &q \wedge \neg r \to p, \end{aligned} \tag{3}$$

corresponding to rules (2). The reduct of each of them with respect to the interpretation $\emptyset$ is the tautology $\bot \to \bot$; since $\emptyset$ is a minimal model of this tautology, it is a stable model of theory (3). The reduct of (3) with respect to $\{p, q\}$ is

$$\begin{aligned} &p \to q, \\ &q \wedge \neg\bot \to p. \end{aligned}$$

The interpretation $\{p, q\}$ is a model of this reduct, but it is not minimal: its subset $\emptyset$ is a model of the reduct as well. Consequently, $\{p, q\}$ is not a stable model of (3).

It is easy to check by induction that an interpretation I satisfies the reduct F^I if and only if it satisfies F. It follows that every stable model of a propositional theory T is a model of T.

It is clear also that every atom occurring in F^I belongs to I.

3 A tale of two graphs

A *nondisjunctive rule* is an implication whose consequent is an atom. Take a set T of nondisjunctive rules. What graph will we designate as the positive dependency graph of T? As far as the set of vertices is concerned, the decision is straightforward – we will include all atoms that occur in the members:

$$Body \to H \tag{4}$$

of T. How will we choose the edges of the graph? For every formula (4) in T, the graph will include edges going from H to some of the atoms occurring in *Body*. But how will we decide which of the atoms occurring in *Body* to choose as the heads of edges?

A subformula of a formula F is called *strictly positive* if it does not belong to the antecedent of any implication. For instance, in a conjunction of literals:

$$B_1 \wedge \cdots \wedge B_m \wedge \neg B_{m+1} \wedge \cdots \neg B_n$$

the atoms $B_1, \dots, B_m$ are strictly positive, and the atoms $B_{m+1}, \dots, B_n$ are not (recall that $\neg B_i$ is shorthand for the implication $B_i \to \bot$). In our more general definition of the positive dependency graph, it would be natural to include, for every member (4) of T, the edges from H to all atoms that have

at least one strictly positive occurrence in *Body*.

We will denote the graph formed from T according to this rule by $G^{sp}(T)$. (The superscript *sp* stands for *strictly positive*.)

However, the publications mentioned in the introduction (Ferraris *et al*. 2006; 2011; Lifschitz and Yang 2013; Ferraris *et al*. 2009; Harrison and Lifschitz 2016) use a different, and more complicated, definition of the positive dependency graph. A subformula of a formula F is called

- *positive* if the number of implications containing it in the antecedent is even, and

https://doi.org/10.1017/S1471068422000333 Published online by Cambridge University Press

- *nonnegated* if it does not belong to the antecedent of any implication with the consequent $\bot$.

The graph designated as the positive dependency graph of T in the publications mentioned above has the same vertices as $G^{sp}(T)$, but its edges go from H to all atoms that have

at least one positive nonnegated occurrence in *Body*

for all members (4) of T. We will denote this graph by $G^{pnn}(T)$. (The superscript *pnn* stands for *positive nonnegated.*) It is clear that $G^{sp}(T)$ is a subgraph of $G^{pnn}(T)$. For example, if T is

$$((p \to q) \to r) \to s$$

then the only edge of $G^{sp}(T)$ is (s, r); $G^{pnn}(T)$ has two edges, (s, r) and (s, p).

4 Which graph is right for your problem?

The definitions of G^{sp} and G^{pnn} in Section 3 are limited to sets of nondisjunctive rules. We will now extend them to arbitrary propositional theories; this generalization will be used in Sections 4.2–4.4.

A strictly positive occurrence of an implication $Body \to Head$ in a formula F is called a *rule* of F. For any propositional theory T, by $G^{sp}(T)$ we denote the directed graph such that

(a) its vertices are the atoms occurring in the members of T, and
(b) for every rule $Body \to Head$ of any member of T, it includes the edge (H, B) for every atom B that has at least one strictly positive occurrence in *Body* and every atom H that has at least one strictly positive occurrence in *Head*.

By $G^{pnn}(T)$ we denote the directed graph satisfying conditions (a) and

(b′) for every rule $Body \to Head$ of any member of T, it includes the edge (H, B) for every atom B that has at least one positive nonnegated occurrence in *Body* and every atom H that has at least one strictly positive occurrence in *Head*.

For any formula F, we will write $G^{sp}(\{F\})$ as $G^{sp}(F)$, and similarly for G^{pnn}.

4.1 Supported models

A model I of a set T of nondisjunctive rules is *supported* if every atom A in I is the consequent of some member $Body \to A$ of T such that I satisfies *Body*. Supported models are important because of their relation to program completion (Clark 1978; Lloyd and Topor 1984): for any finite set T of nondisjunctive rules, an interpretation I is a model of the completion of T if and only if I is a supported model of T (Apt *et al*. 1988).

Every stable model of a set of nondisjunctive rules is supported, but the converse is, generally, not true. For instance, $\{p, q\}$ is a supported model of (3), but it is not stable. From published work on generalizations of Fages' theorem, we know that the stability of all supported models can be asserted for the sets T of nondisjunctive rules such that the graph $G^{pnn}(T)$ has no infinite paths (Lifschitz and Yang 2013, Electronic Appendix B).

(For finite T, this is the same as assuming that the graph is acyclic.) We will show that the graph $G^{sp}(T)$ has the same property:

Theorem 1

For any set T of nondisjunctive rules, if the graph $G^{sp}(T)$ has no infinite paths then every supported model of T is stable.

Thus cycles and other infinite paths in $G^{pnn}(T)$ containing edges that are not included in $G^{sp}(T)$ are harmless – they do not destroy the match between stable models and supported models. For instance, let T be the pair of formulas:

$$\begin{aligned} &p \to q, \\ &((q \to r) \to r) \to p. \end{aligned} \tag{5}$$

The graph $G^{sp}(T)$ has two edges, (q, p) and (p, r), and it is acyclic. Consequently, the stable models of T are identical to its supported models $\emptyset$, $\{p, q\}$. The graph $G^{pnn}(T)$ is not acyclic in this case because of the additional edge (p, q).

4.2 Loops

For any formula F and any set Y of atoms occurring in F, the "negated external support" formula $\mathrm{NES}_F(Y)$ is defined recursively, as follows:

- for an atom A, $\mathrm{NES}_A(Y)$ is $\bot$ if $A \in Y$, and A otherwise;
- $\mathrm{NES}_\bot(Y) = \bot$;
- $\mathrm{NES}_{F \wedge G}(Y) = \mathrm{NES}_F(Y) \wedge \mathrm{NES}_G(Y)$;
- $\mathrm{NES}_{F \vee G}(Y) = \mathrm{NES}_F(Y) \vee \mathrm{NES}_G(Y)$;
- $\mathrm{NES}_{F \to G}(Y) = (\mathrm{NES}_F(Y) \to \mathrm{NES}_G(Y)) \wedge (F \to G)$

(Ferraris *et al*. 2006, Section 3). A set I of atoms occurring in F is a stable model of F iff it satisfies both F and the *loop formulas*:

$$\bigwedge_{A \in Y} (A \to \neg \mathrm{NES}_F(Y)) \tag{6}$$

for all sets Y of atoms occurring in F (Ferraris *et al*. 2006, Theorem 2). Furthermore, according to the same theorem, there is no need to check all loop formulas (6). A set Y of atoms occurring in F is called a *loop* for F if the subgraph of $G^{pnn}(F)$ induced by Y is strongly connected. If I satisfies both F and the loop formulas (6) for all loops Y of F, then I is a stable model of F.

The discussion in Section 4.1 suggests the question: will the last result remain true if we replace the graph $G^{pnn}(F)$ in the definition of a loop by the smaller graph $G^{sp}(F)$? The answer to this question is no. A counterexample is given by the formula:

$$(p \to q) \wedge (((q \to p) \to p) \to p) \tag{7}$$

as F, and $\{p, q\}$ as I. Indeed, the edges of the graph $G^{sp}(F)$ in this case are (q, p) and (p, p), and the sets Y for which the subgraph of $G^{sp}(F)$ induced by Y is strongly connected are $\{p\}$ and $\{q\}$. Calculations show that each of the formulas

$$\mathrm{NES}_F(\{p\}), \ \mathrm{NES}_F(\{q\})$$

is equivalent to $\neg p \wedge \neg q$, so that each of the loop formulas

$$p \to \neg \mathrm{NES}_F(\{p\}),\ q \to \neg \mathrm{NES}_F(\{q\})$$

is a tautology. Thus, I is a model of F that satisfies these loop formulas, although it is not stable.

The graph $G^{pnn}(F)$, on the other hand, has one more edge, (p, q). The subgraph of this graph induced by $\{p, q\}$ is strongly connected, and the corresponding loop formula eliminates the model I.

4.3 Pointwise stable models

Recall that a model I of a propositional theory T is stable if and only if no proper subset of I satisfies the reduct F^I (Section 2). We say that a model I of T is *pointwise stable* if there is no atom A in I such that $I \setminus \{A\}$ satisfies the reduct T^I. For example, $\{p, q\}$ is a pointwise stable model of $p \leftrightarrow q$. Indeed, the reduct of $p \leftrightarrow q$ with respect to $\{p, q\}$ is $p \leftrightarrow q$; it is not satisfied by any of the two sets obtained from $\{p, q\}$ by removing a single atom.

From published work on pointwise stable models (Ferraris *et al.* 2011, Theorem 13), we can conclude that for any finite propositional theory T such that the graph $G^{pnn}(T)$ is acyclic, every pointwise stable model of T is stable. The following theorem shows that the graph $G^{pnn}(T)$ in this statement can be replaced by the smaller graph $G^{sp}(T)$:

Theorem 2
For any propositional theory T, if the graph $G^{sp}(T)$ has no infinite paths then all pointwise stable models of T are stable.

The additional generality of this theorem related to the use of $G^{sp}(T)$ instead of $G^{pnn}(T)$ can be illustrated by formulas (5). Theorem 2 shows that all pointwise stable models of that theory are stable.

4.4 Splitting

Splitting a logic program (Lifschitz and Turner 1994) allows us to relate its stable models to stable models of its parts. The form of splitting described below is a special case of published results on splitting first-order formulas (Ferraris *et al.* 2009) and infinitary propositional theories (Harrison and Lifschitz 2016), expressed in a form convenient for our present purposes.

Let $\{P, Q\}$ be a partition of the set of atoms occurring in a formula $F \wedge G$. If

(i) every atom that has a strictly positive occurrence in F belongs to P, and
(ii) every atom that has a strictly positive occurrence in G belongs to Q, and
(iii) every strongly connected component of $G^{pnn}(F \wedge G)$ is contained in P or in Q,

then any set of atoms is a stable model of $F \wedge G$ if and only if it is a stable model of each of the formulas:

$$F \wedge \bigwedge_{A \in Q}(A \vee \neg A),\ G \wedge \bigwedge_{A \in P}(A \vee \neg A).$$

This assertion will become incorrect, however, if we replace $G^{pnn}(F \wedge G)$ in condition (iii) by $G^{sp}(F \wedge G)$. A counterexample is given by formula (7) as $F \wedge G$, $\{q\}$ as P,

https://doi.org/10.1017/S1471068422000333 Published online by Cambridge University Press

and $\{p\}$ as Q. Indeed, $\{p, q\}$ is a stable model of each of the formulas:

$$(p \rightarrow q) \wedge (p \vee \neg p),$$
$$(((q \rightarrow p) \rightarrow p) \rightarrow p) \wedge (q \vee \neg q),$$

but not a stable model of (7).

5 Proofs of theorems

It is convenient to prove Theorem 2 first.

For any formula F, $\mathrm{SPos}(F)$ stands for the set of atoms that have at least one strictly positive occurrence in F. For any propositional theory T, $\mathrm{SPos}(T)$ is the union of the sets $\mathrm{SPos}(F)$ over all formulas F in T.

Lemma 1

(Lifschitz and Yang 2013, Electronic Appendix C, Lemma F) If an interpretation I satisfies a formula F, then every interpretation J such that $\mathrm{SPos}(F^I) \subseteq J$ satisfies F^I.

Lemma 2

Let F be a propositional formula, let I, J be interpretations such that $J \subset I$, and let M be an atom in $I \setminus J$ such that

$$\text{for every edge } (M, A) \text{ of } G^{sp}(F^I),\ A \in J. \tag{8}$$

If M belongs to $\mathrm{SPos}(F^I)$ and J satisfies F^I, then $I \setminus \{M\}$ satisfies F^I as well.

Proof. Note first that, under the assumptions of the lemma, I satisfies F. Indeed, otherwise F^I would be $\bot$, which contradicts the assumption that J satisfies F^I.

The proof is by structural induction. Formula F is neither an atom nor $\bot$. Indeed, otherwise F^I would be an atom or $\bot$ too; since $M \in \mathrm{SPos}(F^I)$, $F^I = M$. Since $M \in I \setminus J$, this contradicts the assumption that J satisfies F^I.

Let F be $F_1 \wedge F_2$ so that F^I is $F_1^I \wedge F_2^I$. Since J satisfies F^I, J satisfies F_i^I $(i = 1, 2)$. We need to show that $I \setminus \{M\}$ satisfies F_i^I as well. *Case 1:* $M \in \mathrm{SPos}(F_i^I)$. Since every rule of F_i^I is a rule of F^I, $G^{sp}(F_i^I)$ is a subgraph of $G^{sp}(F^I)$; from (8), we can conclude that

$$\text{for every edge } (M, A) \text{ of } G^{sp}(F_i^I),\ A \in J.$$

Then $I \setminus \{M\}$ satisfies F_i^I by the induction hypothesis. *Case 2:* $M \notin \mathrm{SPos}(F_i^I)$. Since $\mathrm{SPos}(F_i^I)$ is a subset of I, it follows that $\mathrm{SPos}(F_i^I) \subseteq I \setminus \{M\}$. On the other hand, I satisfies F_i, because J satisfies F_i^I. By Lemma 1, these two facts imply that $I \setminus \{M\}$ satisfies F_i^I.

If F is $F_1 \vee F_2$, then the proof is similar.

Let F be $F_1 \rightarrow F_2$. Then F^I is $F_1^I \rightarrow F_2^I$ and $\mathrm{SPos}(F^I) = \mathrm{SPos}(F_2^I)$ so that $M \in \mathrm{SPos}(F_2^I)$. It follows that for every atom A in $\mathrm{SPos}(F_1^I)$, the graph $G^{sp}(F^I)$ has an edge from M to A. Hence, by assumption (8), every such atom A belongs to J. Thus,

$$\mathrm{SPos}(F_1^I) \subseteq J. \tag{9}$$

https://doi.org/10.1017/S1471068422000333 Published online by Cambridge University Press

Case 1: J satisfies F_2^I. Since every rule of F_2^I is a rule of F^I, $G^{sp}(F_2^I)$ is a subgraph of $G^{sp}(F^I)$; from (8) we can conclude that

$$\text{for every edge } (M, A) \text{ of } G^{sp}(F_2^I),\ A \in J.$$

By the induction hypothesis, it follows that $I \setminus \{M\}$ satisfies F_2^I, and consequently satisfies F^I. *Case 2:* J does not satisfy F_2^I. Then I does not satisfy F_1. Indeed, otherwise we would be able to conclude by (9) and Lemma 1 that J satisfies F_1^I, which contradicts the assumption that J satisfies F^I. Hence $F_1^I = \bot$, and F^I is a tautology.

Proof of Theorem 2. Let I be a model of T. Assume that J is a proper subset of I that satisfies T^I; we need to show that a subset satisfying T^I can be obtained from I by removing a single atom.

We will show first that the set $I \setminus J$ contains an atom M satisfying condition (8). *Case 1:* $I \setminus J$ contains an atom that is not a vertex of $G^{sp}(T^I)$. Then condition (8) holds for that atom trivially. *Case 2:* all atoms in $I \setminus J$ are vertices of $G^{sp}(T^I)$. Assume that condition (8) is not satisfied for any of the vertices M in $I \setminus J$, so that

$$\text{for every vertex } M \text{ in } I \setminus J,\ G^{sp}(T^I) \text{ has an edge to some vertex } A \text{ in } I \setminus J.$$

Since the set $I \setminus J$ is non-empty, it follows that the graph $G^{sp}(T^I)$ has an infinite path. But this is impossible, because $G^{sp}(T^I)$ is a subgraph of $G^{sp}(T)$.

Take an atom M in $I \setminus J$ that satisfies condition (8), and any formula F from T. If $M \in \mathrm{SPos}(F^I)$, then we conclude that $I \setminus \{M\}$ satisfies F^I by Lemma 2. Otherwise, $\mathrm{SPos}(F^I) \subseteq I \setminus \{M\}$, and $I \setminus \{M\}$ satisfies F^I by Lemma 1.

Proof of Theorem 1. Let I be a supported model of a set T of nondisjunctive rules such that the graph $G^{sp}(T)$ has no infinite paths; we need to show that I is stable. According to Theorem 2, it is sufficient to check that I is pointwise stable.

Take any atom A in I; we need to show that $I \setminus \{A\}$ is not a model of T^I. Since I is supported, T contains a nondisjunctive rule $\mathit{Body} \to A$ such that I satisfies Body. The atom A has no strictly positive occurrences in Body; otherwise, $A, A, \dots$ would be an infinite path in $G^{sp}(T)$. Consequently,

$$\mathrm{SPos}(\mathit{Body}^I) \subseteq \mathrm{SPos}(\mathit{Body}) \subseteq I \setminus \{A\}.$$

By Lemma 1, it follows that $I \setminus \{A\}$ satisfies Body^I. Therefore, $I \setminus \{A\}$ does not satisfy the formula $\mathit{Body}^I \to A$, which belongs to T^I.

6 Conclusion

The earliest use of positive dependency graphs for propositional formulas (Ferraris *et al.* 2006) was related to the study of loops, and introducing the G^{pnn} construction in that context rather than G^{sp} was fully justified, as we saw in Section 4.2. Using G^{pnn} in the theory of splitting was justified as well (Section 4.4). Theorems 1 and 2 show, on the other hand, that G^{sp} would be a better tool for research on completion and on pointwise stable models.

The definitions of G^{sp} and G^{pnn}, as well as Theorems 1 and 2 and their proofs, can be extended to infinitary propositional formulas.

The positive predicate dependency graph of a first-order formula can be defined in two different ways as well, using either the "sp" approach or the "pnn" approach. The dependency graph defined by Bartholomew and Lee 2019 is the sp-style predicate dependency graph for first-order formulas with intensional functions. Theorem 1 is similar to their Theorem 4. It is less general in some ways (no variables and quantifiers, no intensional functions) and more general in other ways (the theory can be infinite and is not required to be in Clark normal form).

References

Apt, K., Blair, H. and Walker, A. 1988. Towards a theory of declarative knowledge. In *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed. Morgan Kaufmann, San Mateo, CA, 89–148.

Bartholomew, M. and Lee, J. 2019. First-order stable model semantics with intensional functions. *Artificial Intelligence 273*, 56–93.

Clark, K. 1978. Negation as failure. In *Logic and Data Bases*, H. Gallaire and J. Minker, Eds. Plenum Press, New York, 293–322.

Fages, F. 1994. Consistency of Clark's completion and existence of stable models. *Journal of Methods of Logic in Computer Science 1*, 51–60.

Fandinno, J. and Lifschitz, V. 2021. Verification of locally tight programs. In *Technical Communications of the Thirty-seventh International Conference on Logic Programming (ICLP'11)*, A. Formisano, Y. Liu, B. Bogaerts, A. Brik, V. Dahl, C. Dodaro, P. Fodor, G. Pozzato, J. Vennekens and N. Zhou, Eds. Electronic Proceedings in Theoretical Computer Science (EPTCS), vol. 345.

Ferraris, P. 2005. Answer sets for propositional theories. In *Proceedings of International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR)*, 119–131.

Ferraris, P., Lee, J. and Lifschitz, V. 2006. A generalization of the Lin-Zhao theorem. *Annals of Mathematics and Artificial Intelligence 47*, 79–101.

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

Ferraris, P., Lee, J., Lifschitz, V. and Palla, R. 2009. Symmetric splitting in the general theory of stable models. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 797–803.

Gelfond, M. and Lifschitz, V. 1988. The stable model semantics for logic programming. In *Proceedings of International Logic Programming Conference and Symposium*, R. Kowalski and K. Bowen, Eds. MIT Press, 1070–1080.

Harrison, A. and Lifschitz, V. 2016. Stable models for infinitary formulas with extensional atoms. *Theory and Practice of Logic Programming 16,* 5-6, 771–786.

Lifschitz, V. 1986. Pointwise circumscription: Preliminary report. In *Proceedings of National Conference on Artificial Intelligence (AAAI)*, 406–410.

Lifschitz, V. 2019. *Answer Set Programming*. Springer, Berlin/Heidelberg, Germany.

Lifschitz, V. and Turner, H. 1994. Splitting a logic program. In *Proceedings of International Conference on Logic Programming (ICLP)*, P. Van Hentenryck, Ed., 23–37.

Lifschitz, V. and Yang, F. 2013. Lloyd-Topor completion and general stable models. *Theory and Practice of Logic Programming 13,* 4–5.

Lin, F. and Zhao, Y. 2004. ASSAT: Computing answer sets of a logic program by SAT solvers. *Artificial Intelligence 157*, 115–137.

Lloyd, J. and Topor, R. 1984. Making Prolog more expressive. *Journal of Logic Programming 1*, 225–240.

Marek, V. and Truszczynski, M. 1999. Stable models and an alternative logic programming paradigm. In *The Logic Programming Paradigm: A 25-Year Perspective*. Springer Verlag, 375–398.

Niemelä, I. 1999. Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence 25*, 241–273.

Oikarinen, E. and Janhunen, T. 2008. Achieving compositionality of the stable model semantics for Smodels programs. *Theory and Practice of Logic Programming 5–6*, 717–761.