Stable Models and Difference Logic

Ilkka Niemelä
Helsinki University of Technology
Laboratory for Theoretical Computer Science
P.O.Box 5400, FI-02015 TKK, Finland
Ilkka.Niemela@tkk.fi

Dedicated to Victor Marek
on his 65th birthday

Abstract

The paper studies the relationship between logic programs with the stable model semantics and difference logic recently considered in the Satisfiability Modulo Theories framework. Characterizations of stable models in terms of level rankings are developed building on simple linear integer constraints allowed in difference logic. Based on a characterization with level rankings a translation is devised which maps a normal program to a difference logic formula capturing stable models of the program as satisfying valuations of the resulting formula. The translation makes it possible to use a solver for difference logic to compute stable models of logic programs.

1 Introduction

Logic programs with the stable model semantics have emerged as an attractive knowledge representation formalism and approach to solving search problems using the *answer set programming* (ASP) paradigm (Marek & Truszczyński 1999; Lifschitz 1999). The basic idea is that a problem is solved by devising a logic program such that the stable models of the program provide the answers to the problem, i.e., solving the problem is reduced to a stable model computation task (Lifschitz 1999; Marek & Truszczyński 1999; Niemelä 1999; Eiter, Gottlob, & Mannila 1997; Buccafurri, Leone, & Rullo 1997).

Classical propositional logic has been used in a similar way to solve search problems in areas like planning and computed aided verification (Kautz & Selman 1992; 1996; Clarke *et al.* 2001). There is a close relationship between stable models of (propositional) logic programs and classical propositional logic. For example, interesting translations from programs to propositional formulas have been developed starting from Clark's work on program completion, early work by Ben-Eliyahu and Dechter (Ben-Eliyahu & Dechter 1994), and more recently on incremental translations based on *loop formulas* (Lin & Zhao 2002; Lierler & Maratea 2004). The work by Janhunen (Janhunen 2004) seems to provide the most compact translation of normal programs to propositional clauses where stable models correspond directly to classical models of the translation. The size of the translation is $\mathcal{O}(m \log n)$ where m is the length of the program and n is the number of atoms in the program. Such translations can be used to implement stable model computation (Lin & Zhao 2002; Lierler & Maratea 2004) using software tools for solving propositional satisfiability (SAT) problems which have advanced very rapidly in recent years (see `http://www.satcompetition.org/`).

Copyright © 2007, authors listed above. All rights reserved.

In recent years extensions of the SAT problem have been studied quite extensively, in particular, in the SMT (Satisfiability Modulo Theories) framework (Bozzano *et al.* 2005; Nieuwenhuis, Oliveras, & Tinelli 2006). In this paper we consider an extension of SAT called difference logic where propositional logic is extended by allowing simple linear constraints of the form $x_i + k \geq x_j$ where x_i, x_j are integer variables and k is an integer constant. Difference logic is interesting because such linear constraints are useful in various applications and because efficient implementation techniques are available in the SMT framework (Nieuwenhuis & Oliveras 2005; Cotton & Maler 2006).

This paper studies the relationships between logic programs with the stable model semantics and difference logic with the aim to pave the way for a more extensive integration of SMT and other constraint satisfaction technique and ASP methods, in line with work started, e.g., in (Baselice, Bonatti, & Gelfond 2005; Mellarkod & Gelfond 2007).

The rest of the paper is organized as follows. In Section 2 we review basic concepts of normal logic programs and the stable model semantics. Section 3 introduces difference logic and Section 4 develops new characterizations of stable models using linear constraints allowed in difference logic. Section 5 then presents a translation of logic programs to difference logic such that stable models of a program correspond to satisfying valuations of its translation.

2 Stable Models

We consider normal propositional logic programs with the stable model semantics (Gelfond & Lifschitz 1988). Such programs consist of normal rules of the form

$$a \leftarrow b_1, \ldots, b_m, \text{not } c_1, \ldots, \text{not } c_n. \qquad (1)$$

where each a, b_i, c_j is a ground atom. Given a rule r of the form (1) we denote the head a by $\mathrm{H}(r)$, the set of body literals $\{b_1, \ldots, b_m, \text{not } c_1, \ldots, \text{not } c_n\}$ by $\mathrm{B}(r)$, the set of positive body atoms $\{b_1, \ldots, b_m\}$ by $\mathrm{B}^+(r)$ and the set of negative body atoms $\{c_1, \ldots, c_n\}$ by $\mathrm{B}^-(r)$.

We write $\mathrm{At}(P)$ for the set of atoms that appear in a program P. Models of a program are subsets of $\mathrm{At}(P)$. A set of atoms M is said to satisfy an atom a if $a \in M$ and a negative literal not a if $a \notin M$ (denoted by $M \models a$ and $M \models \text{not } a$, respectively). A set of atoms M satisfies a set of literals L ($M \models L$) if it satisfies every literal in L. A rule r of the form (1) is satisfied by M if $M \models \mathrm{H}(r)$ holds whenever $M \models \mathrm{B}(r)$ holds. A program P is satisfied by M (M is a model of P) if each rule in P is satisfied by M (denoted $M \models P$).

Stable models of a program are sets of atoms which satisfy all the rules of the program and are justified by the rules. This is captured using the concept of a *reduct*. For a program P and a set of atoms M, the reduct P^M is defined by

$$P^M = \{\mathrm{H}(r) \leftarrow \mathrm{B}^+(r) \mid r \in P, \mathrm{B}^-(r) \cap M = \emptyset\}.$$

The reduct P^M does not contain any negative literals and, hence, has a unique subset minimal set of atoms satisfying it.

Definition 1 *A set of atoms M is a stable model of a program P iff M is the unique minimal set of atoms satisfying P^M.*

We also employ a related concept of a supported model (Apt, Blair, & Walker 1988). A set of atoms M is a *supported model* of a program P iff $M \models P$ and for every atom $a \in M$ there is a rule $r \in P$ such that $\mathrm{H}(r) = a$ and $M \models \mathrm{B}(r)$.

We define for any program P and $I \subseteq \mathrm{At}(P)$, the set of *support rules* $P_I = \{r \in P \mid I \models \mathrm{B}(r)\}$. Any stable model $M \subseteq \mathrm{At}(P)$ of a normal logic program P is also a supported model of P but the converse does not hold in general (Marek & Subrahmanian 1992).

3 Difference Logic

In this paper we study the relationship of normal logic programs and difference logic as studied, e.g., in (Nieuwenhuis & Oliveras 2005). Let $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ be a set of propositional atoms and $\mathcal{X} = \{x_1, x_2, \ldots, x_m\}$ a set of integer variables. The set of atomic formulas of difference logic consists of propositions in $\mathcal{P}$ and linear constraints of the form $x_i + k \geq x_j$ where k is an arbitrary integer constant and $x_i, x_j \in \mathcal{X}$. The set of difference logic formulas $\mathcal{F}$ is the smallest set that contains the atomic formulas and is closed under negation ($\neg$) and conjunction ($\wedge$). The Boolean connectives $\vee, \rightarrow, \leftrightarrow$ can be defined in the usual way in terms of $\neg$ and $\wedge$. For example,

$$(x_1 + 2 \geq x_2) \leftrightarrow (p_1 \rightarrow \neg(x_2 + 2 \geq x_1))$$

is a formula in difference logic.

A valuation τ consists of a pair of functions $\tau_{\mathcal{P}} : \mathcal{P} \rightarrow \{\bot, \top\}$ and $\tau_{\mathcal{X}} : \mathcal{X} \rightarrow \mathbb{Z}$ where $\mathbb{Z}$ is the set of integers. A valuation is extended to all formulas in $\mathcal{F}$ by defining

$$\tau(x_i + k \geq x_j) = \top \text{ iff } \tau_{\mathcal{X}}(x_i) + k \geq \tau_{\mathcal{X}}(x_j)$$

and applying the usual rules for the Boolean connectives. A formula $\phi \in \mathcal{F}$ is satisfied by a valuation τ iff $\tau(\phi) = \top$. For example, given a valuation τ where $\tau_{\mathcal{X}}(x_1) = \tau_{\mathcal{X}}(x_2) = 1, \tau_{\mathcal{P}}(p_1) = \bot$,

$$\tau((x_1 + 2 \geq x_2) \leftrightarrow (p_1 \rightarrow \neg(x_2 + 2 \geq x_1))) = \top.$$

As difference logic contains classical propositional logic as a special case, it is easy to see that given a formula $\phi \in \mathcal{F}$, the satisfiability problem (deciding whether there is a satisfying valuation for ϕ) is NP-complete. See, for example, (Nieuwenhuis & Oliveras 2005; Cotton & Maler 2006) for SMT based techniques for solving the satisfiability problem.

4 New Characterization of Stable Models

In this section we develop a new characterization of stable models where the idea is to use simple linear constraints of the form $x_i + k \geq x_j$ allowed in difference logic.

First, we define the notion of a *level ranking* of a model and show how stable models can be captured in terms of such rankings.

Definition 2 *Let M be a set of atoms and P a normal program. A function* $\mathrm{lr} : M \rightarrow \mathbb{N}$ *is a* level ranking *of M for P iff for each $a \in M$, there is a rule $r \in P_M$ such that $\mathrm{H}(r) = a$ and for every $b \in \mathrm{B}^+(r)$, $\mathrm{lr}(a) - 1 \geq \mathrm{lr}(b)$.*

It turns out that finding a level ranking for a model implies the stability of the model.

Theorem 3 *Let M be a model of a finite normal program P. Then M is a stable model of P iff there is a level ranking of M for P.*

Proof. Let M be a model of P.

($\Rightarrow$) Suppose M is a stable model of P. We can construct a level ranking lr of M for P using the $\mathrm{T}_{P^M}(\cdot)$ operator which is defined for a program P as $\mathrm{T}_P(I) = \{\mathrm{H}(r) \mid r \in P, I \models \mathrm{B}(r)\}$. For this operator define

$$\mathrm{T}_P{\uparrow}0 = \emptyset$$

and for $i = 0, 1, 2, \ldots$

$$\mathrm{T}_P{\uparrow}(i+1) = \mathrm{T}_P(\mathrm{T}_P{\uparrow}i).$$

Now we assign $\mathrm{lr}(a) = i$ iff $a \in \mathrm{T}_{P^M}{\uparrow}i$ but $a \notin \mathrm{T}_{P^M}{\uparrow}(i-1)$. As M is a stable model of P, we know that $M = \mathrm{T}_{P^M}{\uparrow}\omega$ and for each $a \in M$ there is a unique i such that $a \in \mathrm{T}_{P^M}{\uparrow}i$ but $a \notin \mathrm{T}_{P^M}{\uparrow}(i-1)$. Hence, for $a \in M$ there is a rule r of the form (1) such that $a \leftarrow b_1, \ldots, b_m \in P^M$ and $\mathrm{T}_P{\uparrow}(i-1) \models \{b_1, \ldots, b_m\}$ but then $r \in P_M$ and for every b_j, $\mathrm{lr}(b_j) \leq i-1$ and, hence, $\mathrm{lr}(a) - 1 = i - 1 \geq \mathrm{lr}(b_j)$ which implies that lr is a level ranking.

($\Leftarrow$) Suppose there is a level ranking lr of M for P. We show that then M is the minimal model of P^M which implies that M is a stable model of P. As M is a model of P, it is a model of P^M. Assume that there is $M' \subset M$ such that $M' \models P^M$. Then there is an atom $a \in M - M'$ with the smallest level ranking $\mathrm{lr}(a)$ for which there is a rule r in P such that $\mathrm{H}(r) = a$ and $M \models \mathrm{B}(r)$. Now $a \leftarrow b_1, \ldots, b_m \in P^M$ and for every b_i, $\mathrm{lr}(b_i) < \mathrm{lr}(a)$.

Thus, $b_i \in M'$ and $M' \models \{b_1, \dots, b_m\}$ but $M' \not\models a$. This implies that M' is not a model of P^M, a contradiction. Hence, M is the minimal model of P^M which implies that M is a stable model of P. $\square$

A similar characterization of stable models for ground programs has been discussed in (Elkan 1990) and generalized to programs with variables in (Fages 1994) based on well-founded partial orders.

It should be noticed that for a stable model there can be multiple level rankings.

Example 1 *Consider a program*

$$\begin{aligned} p_1 &\leftarrow . \\ p_2 &\leftarrow p_1. \\ p_3 &\leftarrow p_1. \\ p_3 &\leftarrow p_4. \\ p_4 &\leftarrow p_2. \\ p_4 &\leftarrow p_3. \end{aligned}$$

Now a function satisfying $\mathrm{lr}_1(p_i) = i$ *is a level ranking of the model* $M = \{p_1, p_2, \dots, p_4\}$ *but so is* $\mathrm{lr}_2(p_1) = 1, \mathrm{lr}_2(p_2) = \mathrm{lr}_2(p_3) = 2, \mathrm{lr}_2(p_4) = 3$.

We can achieve a tighter connection between rankings and stable models using the notion of a *strong* level ranking.

Definition 4 *Let M be a set of atoms and P a normal program. A function* $\mathrm{lr} : M \to \mathbb{N}$ *is a* strong level ranking *of M for P iff for each $a \in M$ the following conditions hold:*

1. *There is a rule $r \in P_M$ such that $\mathrm{H}(r) = a$ and for every $b \in \mathrm{B}^+(r)$, $\mathrm{lr}(a) - 1 \geq \mathrm{lr}(b)$.*
2. *If there is a rule $r \in P_M$ such that $\mathrm{H}(r) = a$ and $\mathrm{B}^+(r) = \emptyset$, then $\mathrm{lr}(a) = 1$.*
3. *For every rule $r \in P_M$ such that $\mathrm{H}(r) = a$ there is $b \in \mathrm{B}^+(r)$ with $\mathrm{lr}(b) + 1 \geq \mathrm{lr}(a)$.*

Consider again Example 1 where lr_1 is not a strong level ranking of M for P because of the rule $p_3 \leftarrow p_1$, for which $\mathrm{lr}_1(p_1) + 1 = 2 < \mathrm{lr}_1(p_3) = 3$ but where lr_2 is a strong level ranking.

It turns out the each stable model has a strong level ranking. In fact, the construction in the proof of Theorem 3 produces a strong level ranking of a stable model.

Theorem 5 *Let M be a model of a normal program P. Then M is a stable model of P iff there is a strong level ranking of M for P.*

Moreover, strong level rankings are unique.

Proposition 6 *Let M be a model of a finite normal program P. If there is a strong level ranking of M for P, then the ranking is unique.*

It turns out that strong level rankings are closely related to level numberings introduced in (Janhunen 2004).

Lemma 7 *Let M be a model of a normal program P. Given a function* $\mathrm{lr} : M \to \mathbb{N}$ *define for each rule in $r \in P_M$, a number*

$$\mathrm{lr}(r) = max\{\mathrm{lr}(b) \mid b \in \mathrm{B}^+(r)\} + 1. \qquad (2)$$

If lr *is a strong level ranking of M for P, then for each atom $a \in M$ it holds that*

$$\mathrm{lr}(a) = min\{\mathrm{lr}(r) | r \in P_M, \mathrm{H}(r) = a\}.$$

Theorem 8 *Every strong level ranking when extended to rules as in (2) is a level numbering as defined in (Janhunen 2004) and every level numbering as defined in (Janhunen 2004) when restricted to atoms is a strong level ranking.*

In (Janhunen 2004) level numberings are used as the basis of a compact mapping of logic programs to classical propositional formulas in such a way that classical models correspond directly to stable models of the original program. The result above shows that strong level rankings can be used in a similar way but less integer variables (and hence counters) are needed as counters for rules can be eliminated.

5 Translating Logic Programs to Difference Logic

Using the characterization in Theorem 3 we develop a mapping from logic programs to difference logic such that stable models of a program are captured by the valuations of the resulting difference logic formula.

The mapping $\mathrm{T}_{\mathrm{diff}}(P)$ of a logic program P to difference logic consists of two parts: the standard Clark's completion $\mathrm{CC}(P)$ (Clark 1978) of P and ranking constraints $\mathrm{R}(P)$.

The completion $\mathrm{CC}(P)$ consists of the set of following formulas for each atom $a \in \mathrm{At}(P)$:

- if a does not appear as a head of any rule in P, formula $\neg a$ is included.
- Otherwise formula

$$a \leftrightarrow bd_a^1 \vee \cdots \vee bd_a^k$$

  is included for an atom a which has $k \geq 1$ rules of the form (1) and, furthermore, for each such rule (1) a formula

$$bd_a^1 \leftrightarrow b_1 \wedge \cdots \wedge b_m \wedge \neg c_1 \wedge \cdots \wedge \neg c_n$$

  is added where bd_a^is are new atoms.

The ranking constraints $\mathrm{R}(P)$ consist of a set of formulas

$$a \to \bigvee_{i=1}^{k} (bd_a^i \wedge (x_a - 1 \geq x_{b_1}) \wedge \cdots \wedge (x_a - 1 \geq x_{b_m}))$$

for each atom a which has $k \geq 1$ rules of the form (1) in P where x_bs are integer variables associated to each atom b in the program.

The translation $\mathrm{T}_{\mathrm{diff}}(P)$ of a program P to difference logic is the conjunction of the formulas in $\mathrm{CC}(P) \cup \mathrm{R}(P)$.

It turns out that the valuations of $\mathrm{T}_{\mathrm{diff}}(P)$ correspond to stable models of P.

Theorem 9 *(i) If a set of atoms M is a stable model of a finite normal program P, then there is a satisfying valuation τ of $\mathrm{T}_{\mathrm{diff}}(P)$ such that $M = \{a \in \mathrm{At}(P) \mid \tau(a) = \top\}$.*

(ii) If there is a satisfying valuation τ of $\mathrm{T}_{\mathrm{diff}}(P)$, then $M = \{a \in \mathrm{At}(P) \mid \tau(a) = \top\}$ is a stable model of P.

Proof sketch. (i) If M is a stable model of P, then it is a supported model of P and, hence, gives a satisfying valuation $\tau_{\mathcal{P}}$ for the completion $\mathrm{CC}(P)$. By Theorem 3 there is a level ranking lr of M. If we set for every atom $a \in M$, $\tau_{\mathcal{X}}(x_a) = \mathrm{lr}(a)$, then the resulting valuation τ satisfies also

$\mathrm{R}(P)$ and, hence, $\mathrm{T}_{\mathrm{diff}}(P)$ such that $M = \{a \in \mathrm{At}(P) \mid \tau(a) = \top\}$.

(ii) If there is a satisfying valuation τ of $\mathrm{T}_{\mathrm{diff}}(P)$, then $M = \{a \in \mathrm{At}(P) \mid \tau(a) = \top\}$ is a supported model of P as τ satisfies $\mathrm{CC}(P)$. Mapping every atom $a \in M$ with $\mathrm{lr}(a) = \tau_{\mathcal{X}}(x_a)$ gives a level ranking of M for P because τ also satisfies $\mathrm{R}(P)$. Then by Theorem 3 M is a stable model of P. □

Example 2 *Consider a program P:*

$$\begin{array}{l} p \leftarrow q, \mathit{not}\ r. \\ q \leftarrow p, \mathit{not}\ r. \end{array}$$

The completion $\mathrm{CC}(P)$ *is*

$$\begin{array}{l} \neg r \\ p \leftrightarrow bd_p^1 \\ bd_p^1 \leftrightarrow q \wedge \neg r \\ q \leftrightarrow bd_q^1 \\ bd_q^1 \leftrightarrow p \wedge \neg r \end{array}$$

and the ranking constraints $\mathrm{R}(P)$ *are*

$$\begin{array}{l} p \rightarrow (bd_p^1 \wedge (x_p - 1 \geq x_q)) \\ q \rightarrow (bd_q^1 \wedge (x_q - 1 \geq x_p)). \end{array}$$

The translation $\mathrm{T}_{\mathrm{diff}}(P)$ *has a satisfying valuation* τ *where* $\tau_{\mathcal{P}}(p) = \tau(q) = \bot$ *and, hence, P has a stable model* $\{\}$. *Notice that there is no satisfying valuation* τ *where* $\tau_{\mathcal{P}}(p) = \tau(q) = \top$ *because then in that valuation also* $\tau(x_p - 1 \geq x_q) = \tau(x_q - 1 \geq x_p) = \top$ *should hold which is impossible.*

Theorem 9 implies that by employing the translation we can compute stable models using a solver for difference logic.

6 Conclusions

The paper studies the relationship between logic programs with the stable model semantics and difference logic. We devise a characterization of stable models using level rankings and strong level rankings which are based on simple linear integer constraints of the form $x_i + k \geq x_j$ allowed in difference logic. Based of the characterization using level rankings we develop a translation of normal programs to difference logic which captures stable models of a program as satisfying valuations of the resulting difference logic formula.

There are several interesting topics of further research. Strong level rankings could be used as a basis for mapping logic programs to propositional formulas in a similar way as proposed in (Janhunen 2004) but perhaps using less counters as rule counters could be eliminated. The translation to difference logic opens up the possibility of using difference logic solvers to compute stable models and experimental work is needed to evaluate the potential of this approach. The translation makes it possible to embed rule based reasoning directly into SMT systems that support difference logic. This could be a promising way to integrate ASP and SMT techniques.

Acknowledgements

The support of the Academy of Finland through grant 211025 is gratefully acknowledged.

References

Apt, K.; Blair, H.; and Walker, A. 1988. Towards a theory of declarative knowledge. In Minker, J., ed., *Foundations of Deductive Databases and Logic Programming*. Los Altos: Morgan Kaufmann Publishers. 89–148.

Baselice, S.; Bonatti, P. A.; and Gelfond, M. 2005. Towards an integration of answer set and constraint solving. In *Logic Programming, 21st International Conference, ICLP 2005, Sitges, Spain, October 2-5, 2005, Proceedings*, 52–66.

Ben-Eliyahu, R., and Dechter, R. 1994. Propositional semantics for disjunctive logic programs. *Annals of Mathematics and Artificial Intelligence* 12(1-2):53–87.

Bozzano, M.; Bruttomesso, R.; Cimatti, A.; Junttila, T. A.; van Rossum, P.; Schulz, S.; and Sebastiani, R. 2005. Mathsat: Tight integration of sat and mathematical decision procedures. *Journal of Automated Reasoning* 35(1-3):265–293.

Buccafurri, F.; Leone, N.; and Rullo, P. 1997. Strong and weak constraints in disjunctive datalog. In *Proceedings of the 4th International Conference on Logic Programming and Non-Monotonic Reasoning*, 2–17. Springer-Verlag.

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

Clarke, E.; Biere, A.; Raimi, R.; and Zhu, Y. 2001. Bounded model checking using satisfiability solving. *Formal Methods in System Design* 19(1):7–34.

Cotton, S., and Maler, O. 2006. Fast and flexible difference constraint propagation for dpll(t). In *Theory and Applications of Satisfiability Testing - SAT 2006, 9th International Conference, Seattle, WA, USA, August 12-15, 2006, Proceedings*, 170–183.

Eiter, T.; Gottlob, G.; and Mannila, H. 1997. Disjunctive datalog. *ACM Transactions on Database Systems* 22(3):364–418.

Elkan, C. 1990. A rational reconstruction of nonmonotonic truth maintenance systems. *Artificial Intelligence* 43:219–234.

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

Gelfond, M., and Lifschitz, V. 1988. The stable model semantics for logic programming. In *Proceedings of the 5th International Conference on Logic Programming*, 1070–1080. Seattle, USA: The MIT Press.

Janhunen, T. 2004. Representing normal programs with clauses. In *Proceedings of the 16th European Conference on Artificial Intelligence*, 358–.362. IOS Press.

Kautz, H., and Selman, B. 1992. Planning as satisfiability. In *Proceedings of the 10th European Conference on Artificial Intelligence*, 359 – 363.

Kautz, H., and Selman, B. 1996. Pushing the envelope: Planning, propositional logic, and stochastic search. In *Proceedings of the 13th National Conference on Artificial Intelligence*, 1194–1201.

Lierler, Y., and Maratea, M. 2004. Cmodels-2: SAT-based answer set solver enhanced to non-tight programs. In *Logic Programming and Nonmonotonic Reasoning, 7th International Conference, LPNMR 2004, Fort Lauderdale, FL, USA, January 6-8, 2004, Proceedings*, 346–350.

Lifschitz, V. 1999. Answer set planning. In *Proceedings of the 16th International Conference on Logic Programming*, 25–37. Las Cruces, New Mexico: The MIT Press.

Lin, F., and Zhao, Y. 2002. ASSAT: Computing answer sets of a logic program by SAT solvers. In *Proceedings of the 18th National Conference on Artificial Intelligence*, 112–117. Edmonton, Alberta, Canada: The AAAI Press.

Marek, V., and Subrahmanian, V. 1992. The relationship between stable, supported, default and autoepistemic semantics for general logic programs. *Theoretical Computer Science* 103:365–386.

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

Mellarkod, V. S., and Gelfond, M. 2007. Enhancing ASP systems for planning with temporal constraints. In *Proceedings of the 9th International Conference on Logic Programming and Nonmonotonic Reasoning*, 309–314.

Niemelä, I. 1999. Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence* 25(3,4):241–273.

Nieuwenhuis, R., and Oliveras, A. 2005. DPLL(T) with exhaustive theory propagation and its application to difference logic. In *Computer Aided Verification, 17th International Conference, CAV 2005, Edinburgh, Scotland, UK, July 6-10, 2005, Proceedings*, 321–334.

Nieuwenhuis, R.; Oliveras, A.; and Tinelli, C. 2006. Solving sat and sat modulo theories: From an abstract davis–putnam–logemann–loveland procedure to DPLL(T). *Journal of the ACM* 53(6):937–977.