Safe Formulas in the General Theory of Stable Models (Preliminary Report)

Joohyung Lee[1], Vladimir Lifschitz[2], and Ravi Palla[1]

[1]School of Computing and Informatics, Arizona State University, USA
[2]Department of Computer Sciences, University of Texas at Austin, USA
[1]{joolee,Ravi.Palla}@asu.edu, [2]vl@cs.utexas.edu

Abstract. Safe first-order formulas generalize the concept of a safe rule, which plays an important role in the design of answer set solvers. We show that any safe sentence is equivalent, in a certain sense, to the result of its grounding—to the variable-free sentence obtained from it by replacing all quantifiers with multiple conjunctions and disjunctions. It follows that a safe sentence and the result of its grounding have the same stable models, and that stable models of a safe sentence can be characterized by a formula of a simple syntactic form.

1 Introduction

The definition of a stable model proposed in [1] is more general than the original definition from [2]: it applies to models of arbitrary first-order sentences. Logic programs referred to in the 1988 definition are identified in this theory with first-order formulas of a special form. For instance, the rule

$$p(x) \leftarrow not\ q(x) \tag{1}$$

is treated as alternative notation for the sentence

$$\forall x(\neg q(x) \rightarrow p(x)). \tag{2}$$

In this example, stable models are the interpretations of the unary predicate constants p and q (in the sense of first-order logic) that make p identically true and q identically false.

This general definition of a stable model involves a syntactic transformation of formulas, which is similar to the circumscription operator [3]—it turns a first-order sentence into a stronger second-order sentence. There is an important difference, however, between stable models and models of circumscription. Two sentences may be equivalent (that is, have the same models), but have different *stable* models. For instance, formula (2) is equivalent to

$$\forall x(\neg p(x) \rightarrow q(x)),$$

but the stable models of these two formulas are not the same. The equivalent transformations of formulas that preserve their stable models are studied in [4].

They are represented there by a subsystem of classical logic called **SQHT**$^=$ ("static quantified logic of here-and-there with equality"). This deductive system includes all axioms and inference rules of intuitionistic logic with equality, the decidable equality axiom

$$x = y \vee x \neq y \tag{3}$$

and two other axiom schemas, but it does not include the general law of the excluded middle $F \vee \neg F$.

In [5], the new approach to stable models is used to define the semantics of an answer set programming language with choice rules and counting, called RASPL-1. The meaning of a RASPL-1 program is defined in terms of the stable models of a first-order sentence associated with the program, which is called its "FOL-representation." For instance, the FOL-representation of the RASPL-1 rule

$$p \leftarrow \{x : q(x)\}\, 1 \tag{4}$$

is the formula

$$\neg\exists xy(q(x) \wedge q(y) \wedge x \neq y) \rightarrow p. \tag{5}$$

In this note, we continue one line of research from [5], the study of safe sentences and their stable models. The definition of a safe sentence, reproduced in the next section, is related to some ideas of [6].[1] It extends the familiar concept of a safe rule, which plays an important role in the design of answer set solvers [7, Section 2.1]. For instance, rule (1) is not safe, and for this reason it is not allowed in the input of any of the existing systems for computing stable models. Rule (4) is safe, and we expect that it will be accepted by a future implementation of RASPL-1.

According to Proposition 1 below, stable models of a safe sentence (without function symbols) have what can be called the "small predicate property": the relation represented by any of its predicate constants can hold for a tuple of arguments only if each member of the tuple is represented by an object constant. We show, furthermore, that any safe sentence is equivalent, in a certain sense, to the result of its grounding—to the variable-free sentence obtained from it by replacing all quantifiers with multiple conjunctions and disjunctions (Proposition 2). We derive from these two facts that a safe sentence and the result of its grounding have the same stable models (Proposition 3). This theorem leads us to the conclusion that stable models of a safe sentence can be characterized by a sentence of a simple syntactic structure—not just first-order, but universal and, moreover, "almost variable-free" (Proposition 4).

[1] Topor and Sonenberg [6] defined the notion of "allowed" formulas, similar to the notion of safe formulas, in a much more limited setting of stratified deductive databases. (That paper was written before the invention of the stable model semantics.) The definitions are not equivalent to each other. For example, $\exists x(\neg p(x) \rightarrow q)$ is safe but not allowed; $\exists x(\neg p(x) \rightarrow q(x))$ is allowed but not safe.

2 Review: Safe Sentences

We consider first-order formulas that may contain object constants and equality but no function constants of arity > 0. $\neg F$ is shorthand for $F \to \bot$, $F \leftrightarrow G$ is shorthand for $(F \to G) \wedge (G \to F)$, and $\top$ is shorthand for $\bot \to \bot$. A *sentence* is a formula without free variables.

Recall that a traditional rule—an implication of the form

$$(L_1 \wedge \cdots \wedge L_n) \to A, \tag{6}$$

not containing equality, where $L_1, \ldots, L_n$ are literals and A is an atom—is considered safe if every variable occurring in it occurs in one of the positive literals in the antecedent. The definition of a safe formula from [5], reproduced below, generalizes this condition to arbitrary sentences in prenex form. The assumption that the formula is in prenex form is not a significant limitation in the general theory of stable models, because all steps involved in the standard process of converting a formula to prenex form are equivalent transformations in **SQHT**$^=$ [8].

To every quantifier-free formula F we assign a set $\mathrm{RV}(F)$ of its *restricted variables* as follows:

- For an atomic formula F,
  - if F is an equality between two variables then $\mathrm{RV}(F) = \emptyset$;
  - otherwise, $\mathrm{RV}(F)$ is the set of all variables occurring in F;
- $\mathrm{RV}(\bot) = \emptyset$;
- $\mathrm{RV}(F \wedge G) = \mathrm{RV}(F) \cup \mathrm{RV}(G)$;
- $\mathrm{RV}(F \vee G) = \mathrm{RV}(F) \cap \mathrm{RV}(G)$;
- $\mathrm{RV}(F \to G) = \emptyset$.

It is clear, for instance, that a variable is restricted in the antecedent of (6) iff it occurs in one of the positive literals among $L_1, \ldots, L_n$.

Consider a sentence F in prenex form: $Q_1 x_1 \cdots Q_n x_n M$ (each Q_i is $\forall$ or $\exists$; $x_1, \ldots, x_n$ are distinct variables; the matrix M is quantifier-free). We say that F is *safe* if every occurrence of each of the variables x_i in M is contained in a subformula $G \to H$ that satisfies two conditions:

(a) the subformula is positive in M if Q_i is $\forall$, and negative in M if Q_i is $\exists$;
(b) x_i is restricted in G.

3 Properties of Safe Sentences

We assume that the reader is familiar with the definition of the stable model operator SM from [1]. Proposition 1 below shows that all stable models of a safe sentence have the small predicate property: the relation represented by any of its predicate constants p_i can hold for a tuple of arguments only if each member of the tuple is represented by an object constant occurring in F. To make this

idea precise, we will use the following notation: for any finite set $\mathbf{c}$ of object constants, $in_{\mathbf{c}}(x_1, \ldots, x_m)$ stands for the formula

$$\bigwedge_{1 \leq j \leq m} \bigvee_{c \in \mathbf{c}} x_j = c.$$

The small predicate property can be expressed by the conjunction of the sentences

$$\forall \mathbf{x}(p_i(\mathbf{x}) \to in_{\mathbf{c}}(\mathbf{x}))$$

for all predicate constants p_i occurring in F, where $\mathbf{x}$ is a list of distinct variables. We will denote this sentence by $SPP_{\mathbf{c}}$. By $c(F)$ we denote the set of all object constants occurring in F.

Proposition 1 *For any safe sentence F,* SM$[F]$ *entails* $SPP_{c(F)}$.

Corollary 1 *For any safe sentence F that does not contain object constants,* SM$[F]$ *entails the formulas* $\forall \mathbf{x} \neg p_i(\mathbf{x})$ *for all predicate constants p_i of arity > 0.*

Indeed, $SPP_{\emptyset}$ is equivalent to the conjunction of all these formulas.

The process of grounding replaces quantifiers by multiple conjunctions and disjunctions. To make this idea precise, we define, for any sentence F in prenex form and any nonempty finite set $\mathbf{c}$ of object constants, the variable-free formula $\text{Ground}_{\mathbf{c}}[F]$ as follows. If F is quantifier-free then $\text{Ground}_{\mathbf{c}}[F] = F$. Otherwise,

$$\text{Ground}_{\mathbf{c}}[\forall x F(x)] = \bigwedge_{c \in \mathbf{c}} \text{Ground}_{\mathbf{c}}[F(c)],$$

$$\text{Ground}_{\mathbf{c}}[\exists x F(x)] = \bigvee_{c \in \mathbf{c}} \text{Ground}_{\mathbf{c}}[F(c)].$$

As in [4], by $\mathbf{INT}^{=}$ we denote intuitionistic predicate logic with equality, and DE stands for the decidable equality axiom (3). The importance of the logical system $\mathbf{INT}^{=} + \text{DE}$ is determined by the fact that it is a part of $\mathbf{SQHT}^{=}$, so that the provability of a sentence $F \leftrightarrow G$ in this system implies that SM$[F]$ is equivalent to SM$[G]$.

Proposition 2 *For any safe sentence F and any nonempty finite set $\mathbf{c}$ of object constants containing $c(F)$, the equivalence*

$$\text{Ground}_{\mathbf{c}}[F] \leftrightarrow F$$

is derivable from $SPP_{\mathbf{c}}$ in $\mathbf{INT}^{=} + \text{DE}$.

Using Proposition 2 we can prove that the variable-free formula obtained by grounding a safe sentence F has the same stable models as F:

Proposition 3 *For any safe sentence F and any nonempty finite set $\mathbf{c}$ of object constants containing $c(F)$,* SM$[\text{Ground}_{\mathbf{c}}[F]]$ *is equivalent to* SM$[F]$.

In general, the second-order definition of a stable model cannot be expressed in first-order logic. The following theorem shows, however, that in the case of a safe sentence, stable models can be characterized by a very simple first-order formula, almost variable-free:

Proposition 4 *For every safe sentence F there exists a variable-free formula G such that* $\mathrm{SM}[F]$ *is equivalent to* $G \wedge SPP_{c(F)}$.

4 Conclusion

In this paper we investigated properties of stable models of safe formulas in a semantically general situation, not limited to Herbrand models, and established a few positive results. We saw, in particular, that grounding a safe sentence preserves its stable models even in this general case, and that the stable models of a safe sentence can be characterized in first-order logic. We hope that these theorems will help us in future work on non-Herbrand answer set programming.

Acknowledgements

We are grateful to Paolo Ferraris and the anonymous referees for useful comments on the draft of this paper. The first and the third author were partially supported by the National Science Foundation under Grant IIS-0839821. The second author was partially supported by the National Science Foundation under Grant IIS-0712113.

References

1. Ferraris, P., Lee, J., Lifschitz, V.: A new perspective on stable models. In: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI). (2007) 372–379
2. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In Kowalski, R., Bowen, K., eds.: Proceedings of International Logic Programming Conference and Symposium, MIT Press (1988) 1070–1080
3. McCarthy, J.: Circumscription—a form of non-monotonic reasoning. Artificial Intelligence **13** (1980) 27–39,171–172
4. Lifschitz, V., Pearce, D., Valverde, A.: A characterization of strong equivalence for logic programs with variables. In: Procedings of International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR). (2007)
5. Lee, J., Lifschitz, V., Palla, R.: A reductive semantics for counting and choice in answer set programming. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). (2008) To appear.
6. Topor, R.W., Sonenberg, E.A.: On domain independent databases. In Minker, J., ed.: Foundations of Deductive Databases and Logic Programming. Morgan Kaufmann, San Mateo, CA (1988) 217–240
7. Leone, N., Faber, W., Pfeifer, G., Eiter, T., Gottlob, G., Perri, S., Scarcello, F.: The DLV system for knowledge representation and reasoning. ACM Transactions on Computational Logic **7** (2006) 499–562
8. Lee, J., Palla, R.: Yet another proof of the strong equivalence between propositional theories and logic programs. In: Working Notes of the Workshop on Correspondence and Equivalence for Nonmonotonic Theories. (2007)