Translating Preferred Answer Set Programs to Propositional Logic

Vernon Asuncion and Yan Zhang

Intelligent Systems Laboratory
School of Computing and Mathematics
University of Western Sydney, Australia
E-mail: {vernon,yan}@scm.uws.edu.au

Abstract. We consider the problem of whether a given preferred answer set program can be reduced to a propositional formula. Research on this topic is of both theoretical and practical interests: on one hand, it will shed new insights to understand the expressive power of preferred answer set programs; on the other hand, it may also lead to efficient implementations for computing preferred answer sets of logic programs. In this paper, we focus on Brewka and Eiter's preferred answer set programs. We propose a translation from preferred answer set programs to propositional logic and show that there is one-to-one correspondence between the preferred answer sets of the program to the models of the resulting propositional theory. We then link this result to Brewka and Eiter's weakly preferred answer set semantics.

Key words: answer set semantics, prioritized logic programs, answer set computations.

1 Introduction

In recent years, Answer Set Programming (ASP) has become one of the most effective approaches for declarative problem solving in knowledge representation and reasoning. One important research in this area is to translate various answer set programs, such as normal logic programs and disjunctive logic programs, into propositional logic, so that the answer sets of these logic programs are precisely captured by the models of corresponding propositional theories, e.g. [5, 7]. Research on this topic is of both theoretical and practical interests because it has not only provided new insights for a better understanding of the expressive power of answer set programming, but also been led to some efficient computations for answer set programming [7].

On the other hand, preferred answer set programming is a promising method for dealing with conflict resolution in nonmonotonic reasoning. Over the years, a number of various preferred answer set program (also called prioritized logic program) frameworks have been developed, which extended traditional answer set semantics by integrating proper priorities into the underlying logic programs, e.g. [3, 4, 6, 8].

However, it remains as an unaddressed question whether a similar translation can be achieved between preferred answer set programs and propositional logic. This paper provides a positive answer to this question. We focus on Brewka and Eiter's preferred

answer set programs and propose a translation between preferred answer set programs and propositional logic. In particular, we prove that given a preferred logic program, the models of a propositional theory, which consists of the completion, loop formulas, and preference formula of the program, precisely are the same as the preferred answer sets of the underlying program.

The rest of the paper is organized as follows. Section 2 overviews the basic notions and concepts of Brewka and Eiter's preferred and weakly preferred answer set programs. Section 3 presents our translation from the preferred answer set program to propositional logic and proves a one-to-one correspondence between the preferred answer sets of the program and the models of the translated propositional theory. Section 4 then links this result to weakly preferred answer set programs. Finally, section 5 concludes the paper with some discussions.

2 Brewka and Eiter's Preferred Answer Set Semantics: An Overview

Consider a propositional language $\mathcal{L}$ which consists of a set of propositional atoms. A *fully prioritized logic program* on the language $\mathcal{L}$ is a pair $(\Pi, <)$, where Π is a (finite) normal logic program and $<$ is a strict *total order* on Π. Since $<$ is a total order on the set Π, the rules in Π corresponds to a unique ordinal number, and thus to an enumeration $r_1, \cdots, r_\alpha, \cdots, r_{|\Pi|}$ of the elements of Π. Therefore we use the notion $\{r_\alpha\}_<$ to represent $(\Pi, <)$.

A ground rule r is *defeated* by a set of atoms S if there exist some atom $a \in S$ such that a appears in the negative body of r, i.e., "$not\ a$" is a part of r's body. We use $Head(r)$, $Pos(r)$ and $Neg(r)$ to denote the head atom, the set of atoms occurring in the positive body, and the set of atoms occurring in the negative body of rule r, respectively. Given a set of atoms S and a rule r, if $Head(r) \notin S$ and $Pos(r) \subseteq S$ then we refer to r as a *zombie* rule with respect to S or simply "zombie rule" when it is clear from the context. Intuitively, a zombie rule r is a rule which is assured to be *non-generating* with respect to S as $Head(r) \notin S$.

Definition 1. *[2] Let $\Pi_< = (\Pi, <)$ be a fully prioritized grounded (normal) logic program and X a set of ground atoms. Let ${}^X\Pi_{<'} = ({}^X\Pi, <')$ be the fully grounded prioritized logic program such that ${}^X\Pi$ is the set of rules obtained from Π by*

1. *deleting every rules having an atom p in its positive body where $p \notin X$, and*
2. *removing from each remaining rules their positive body,*

and $<'$ is inherited from $<$ by the map $f : {}^X\Pi \longrightarrow \Pi$, i.e., $r'_1 <' r'_2$ iff $f(r'_1) < f(r'_2)$, where $f(r') = r$ is the first rule in Π with respect to $<$ such that r' results from r by step 2.

Note that $<'$ is also a strict total order on ${}^X\Pi$. For a fully prioritized program $\Pi_<$, a set of atoms S, and ${}^S\Pi$ defined as in above, the preferred answer set semantics of $\Pi_<$ is defined through an operator $C_{{}^S\Pi_{<'}} : 2^{Atoms({}^S\Pi_{<'})} \longrightarrow 2^{Atoms({}^S\Pi_{<'})}$[1] such that

[1] Here $Atoms({}^S\Pi_{<'})$ denotes the set of all atoms occurring in ${}^S\Pi_{<'}$.

an answer set A of Π satisfies the priorities if and only if $C_{A_{\Pi_{<'}}}(A) = A$. The formal definition is given below.

Definition 2. *[2] For a fully prioritized grounded (normal) logic program $\Pi_< = (\Pi, <)$ and a set S of atoms, let ${}^S\Pi_{<'} = ({}^S\Pi, <') = \{r_\alpha\}_{<'}$ where ${}^S\Pi$ and $<'$ is defined as in Definition 1. The sequence S_α is defined as follows:*

$$S_0 = \emptyset$$

for $\alpha = 0$, and

$$S_{\alpha+1} = \begin{cases} S_\alpha & \text{if } r_{\alpha+1} \text{ is defeated by } S_\alpha \text{ or} \\ & Head(r_{\alpha+1}) \in S \text{ and } r_{\alpha+1} \text{ defeated by } S, \\ S_\alpha \cup \{Head(r_{\alpha+1})\}, & \text{otherwise.} \end{cases}$$

for $0 < \alpha < |\Pi|$. Then $C_{S_{\Pi_{<'}}}(S) = S_{|\Pi|}$.

For a fully prioritized grounded logic program $\Pi_< = (\Pi, <)$ and an answer set A of Π, A is a *preferred answer set* of $\Pi_<$ if and only if $C_{A_{\Pi_{<'}}}(A) = A$.

Obviously, there are some fully prioritized grounded logic programs that may have no preferred answer set. In [2], this problem was addressed by a proposed relaxation that gives preferred answer sets whenever they exist and an approximation called *weakly preferred answer sets*, in the other cases. For a formal definition, the notion of *inversion* is first introduced.

Definition 3. *[2] Let $<_1$ and $<_2$ be two well-orderings on set S. We define $Inv_S(<_1, <_2)$ (inversions of $<_2$ in $<_1$) as*

$$Inv_S(<_1, <_2) = \{(b, a) \mid a, b \in S, a <_2 b, b <_1 a\}.$$

The idea behind weakly preferred answer sets is linked to counting those inversions from a full prioritization $<_1$ of a grounded logic program Π to another full prioritization $<_2$ of the program. To formally define this, the notion of distance is introduced.

Definition 4. *[2] Let $<_1$ and $<_2$ be well-orderings of a finite set S. The distance from $<_1$ to $<_2$, denoted $d_S(<_1, <_2)$, is defined as*

$$d_S(<_1, <_2) = |Inv_S(<_2, <_1)|.$$

From the above definition, the notion of *preference violation degree,* denoted pvd is defined as follows.

Definition 5. *[2] Let $\Pi_< = (\Pi, <)$ be a finite fully prioritized grounded (normal) logic program. For an answer set A of Π, define $pvd_{\Pi_<}(A)$ (preferrence violation degree of A in $\Pi_<$) as*

$$pvd_{\Pi_<}(A) = min\{d_\Pi(<, <') \mid <' \text{ is any full prioritization of } \Pi \text{ such that } A \text{ is a preferred answer set of } \Pi_{<'} = (\Pi, <')\}.$$

Intuitively, $pvd_{\Pi_<}(A)$ is the minimum distance possible from the full prioritization of $\Pi_<$ to any fully prioritized rule base $\Pi_{<'} = (\Pi, <')$ such that A is a preferred answer set of $\Pi_{<'}$. From the above definitions, the semantics of weakly preferred programs can be formally defined as follows.

Definition 6. *[2] Let $\Pi_< = (\Pi, <)$ be a finite fully prioritized grounded (normal) logic program. We define*

$$pvd(\Pi_<) = min\{pvd_{(\Pi_<)}(A) \mid A \text{ is an answer set of } \Pi\}.$$

Then A is a weakly preferred answer set *of $\Pi_<$ iff $pvd_{\Pi_<}(A) = pvd(\Pi_<)$.*

Informally A is a weakly preferred answer set of a fully prioritized grounded (normal) logic program $\Pi_<$ if there exist a full prioritization $<_1$ of Π such that A is a preferred answer set of $\Pi_{<_1}$, and for any other full prioritization $<_2$ of Π where there exist a preferred answer set A' of $\Pi_{<_2}$, we have $d_\Pi(<, <_1) \leq d_\Pi(<, <_2)$.

3 Preference Formulas and the Translation

In this section, we propose a translation from the preferred answer set semantics to propositional logic, such that a one-to-one correspondence exist for this translation.

Definition 7. *For a finite fully prioritized grounded (normal) logic program $\Pi_< = (\Pi, <)$, we define the* preference formula $PF(\Pi_<)$ *of $\Pi_<$ as follows:*

$$PF(\Pi_<) = \bigwedge_{r \in \Pi_<, Head(r) = a} (\neg a \wedge \bigwedge_{b \in Pos(r)} b \supset \bigvee_{r' \in def_\Pi(r), r' < r} (\bigwedge_{c \in Pos(r')} c \wedge \bigwedge_{d \in Neg(r')} \neg d))$$

where $def_\Pi(r) = \{r' \mid r' \in \Pi, Head(r') \in Neg(r), Head(r') \neq Head(r)$ or $Neg(r') \neq Neg(r)\}$.

Informally, $def_\Pi(r)$ is the set of rules in Π that defeat r in a sense that each rule r' in $def_\Pi(r)$ is different from r with regards to either $Head(r)$ or $Neg(r)$, and $Head(r') \in Neg(r)$.

Theorem 1. *For a finite fully prioritized grounded (normal) logic program $\Pi_< = (\Pi, <)$ and an answer set A of Π, $A \models PF(\Pi_<)$ iff A is a preferred answer set of $\Pi_<$.*

In [7], Lin and Zhao proposed a translation of finite normal logic programs to propositional formulas without the need of extra variables. The translation is of the form $Comp(\Pi) \wedge LF(\Pi)$ where $Comp(\Pi)$ is the completion of the logic program Π and $LF(\Pi)$ is the conjunction of all loop formulas associated with Π. The loop formulas are a way of strengthening the completion of Π such that a set of atoms A is an answer set of Π iff A is a model of $Comp(\Pi) \wedge LF(\Pi)$. Using Lin and Zhao's result, we are able to translate prioritized normal logic programs to propositional formulas via the following theorem. Moreover, the models of the resulting propositional formula are in a one-to-one correspondence with the preferred answer sets of the logic program.

Theorem 2. *For a finite fully prioritized grounded (normal) logic program* $\Pi_< = (\Pi, <)$, $A \models Comp(\Pi) \wedge LF(\Pi) \wedge PF(\Pi_<)$ *iff* A *is a preferred answer set of* $\Pi_<$.

4 Linking Weakly Preferred Answer Set Programs

We now try to link the semantics of *weakly preferred answer sets* by extending our previously defined preference formula. Intuitively, we model the weakly preferred criterion by encoding the inversions between two full prioritizations of the rules of a given program. To achieve this, we introduce two classes of new atoms of the form (r_1, r_2) and $X_{(r_1,r_2)}$ where r_1 and r_2 are rule names of the given program.

Definition 8. *For a finite fully prioritized grounded (normal) logic program* $\Pi_< = (\Pi, <)$*, we define the* weak preference formula $WPF(\Pi_<)$ *as follows:*

$$WPF(\Pi_<) =$$

$$\bigwedge_{r\in\Pi, Head(r)=a} (\neg a \wedge \bigwedge_{b\in Pos(r)} b \supset \bigvee_{r'\in def_\Pi(r)} (\bigwedge_{c\in Pos(r')} c \wedge \bigwedge_{d\in Neg(r')} \neg d \wedge (r', r))) \tag{1}$$

$$\wedge \bigwedge_{r_1,r_2\in\Pi, r_1\neq r_2} (((r_1, r_2) \vee (r_2, r_1)) \wedge ((r_1, r_2) \supset \neg(r_2, r_1))) \tag{2}$$

$$\wedge \bigwedge_{r_1,r_2,r_3\in\Pi, r_1\neq r_2\neq r_3} ((r_1, r_2) \wedge (r_2, r_3) \supset (r_1, r_3)) \tag{3}$$

$$\wedge \bigwedge_{r_1,r_2\in\Pi_<, r_2<r_1} (((r_1, r_2) \supset X_{(r_1,r_2)}) \wedge (X_{(r_1,r_2)} \supset (r_1, r_2))) \tag{4}$$

The formula $WPF(\Pi_<)$ is a conjunction of the four subformulas (1), (2), (3), and (4). As can be seen, formula (1) is similar to $PF(\Pi_<)$ except that a rule that defeats a zombie rule does not necessarily have to be more preferred with respect to $<$ and that if a rule r' defeats a zombie rule r then the atom (r', r) (i.e. encodes r' is more preferred than r but not necessarily with respect to $<$) should be satisfied (i.e. in the model satisfying $WPF(\Pi_<)$). Basically, the conjunction of the two formulas (2) and (3) encodes a full prioritization of the rules in Π (not necessarily $<$) where the prioritization relations are represented by the atoms of the form (r_1, r_2). The last formula (4) encodes the inversions of $<$ from the *other* full prioritization relations (that are represented by the atoms (r_1, r_2)), which are then represented by the atoms of the form $X_{(r_1,r_2)}$ (i.e. it indicates that (r_1, r_2) is an inversion of $r_2 < r_1$).

Before we present our main theorem of this section, we need to first introduce a useful notion. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two propositional languages and $\mathcal{L}_1 \subseteq \mathcal{L}_2$, and A an interpretation of $\mathcal{L}_1$ (i.e. a subset of atoms of $\mathcal{L}_1$), an interpretation B of $\mathcal{L}_2$ is called an *extension* of A on $\mathcal{L}_2$, denoted as $B = ext(A)_{\mathcal{L}_2}$, if $A \subseteq B$, and A and B agree on the truth values of all propositional atoms of $\mathcal{L}_1$.[2]

[2] Intuitively, $\mathcal{L}_2$ is a *superset* of $\mathcal{L}_1$.

Theorem 3. *For a finite fully prioritized grounded (normal) logic program* $\Pi_< = (\Pi, <)$ *and an interpretation* A *of language* $\mathcal{L} = Atom(\Pi)$*,* A *is a weakly preferred answer set of* $\Pi_<$ *iff there exist an extension* $ext(A)_P$ *of* A*, where* $P = Atoms(\Pi) \cup \{(r_i, r_j) \mid r_i, r_j \in \Pi\} \cup \{X_{(r_i,r_j)} \mid r_i, r_j \in \Pi_<, r_j < r_i\}$[3]*, such that* $ext(A)_P \models Comp(\Pi) \wedge LF(\Pi) \wedge WPF(\Pi_<)$ *and for all models* M *of* $Comp(\Pi) \wedge LF(\Pi) \wedge WPF(\Pi_<)$*,* $|ext(A)_P \restriction_X| \leq |M \restriction_X|$.

5 Conclusions

In this paper, we have proposed a translations between Brewka and Eiter's preferred answer set programs and propositional logic. We have also proved a one-to-one correspondence theorem for the translation. Moreover, we also provided a link between the weakly preferred answer sets and propositional logic. We believe that our work will be of practical values to serve as an alternative approach for current preferred answer set programming implementations [6]. Currently we are considering to implement a SAT based preferred answer set solver based on the work developed in this paper. From an implementation viewpoint, since our defined preference and weak preference formulas remain in a polynomial size of the underlying program, techniques of ASSAT [7] may be used to optimize the computation of preferred answer sets. For future work, we consider the possibility of applying similar methods to capture the preferred answer set framework in [3] which allows the specification of *dynamic* orderings such that *static* orderings are a trivial restriction of the more general dynamic case.

References

1. R. Ben-Eliyahu, R. Dechter, Propositional semantics for disjunctive logic programs. *Annals of Mathematics and Artificial Intelligence* 12 (1994) 53-87.
2. G. Brewka and T. Eiter, Preferred answer sets for extended logic programs. *Artificial Intelligence* 109 (1999) 297-356.
3. J. Delgrande, T. Schaub, H. Tompits, A framework for compiling preferences in logic programs. *Theory and Practice of Logic Programming* 3 (2003) 129 - 187.
4. J. Delgrande, T. Schaub, H. Tompits, K. Wang, A classification and survey of preference handling approaches in nonmonotonic reasoning. *Computational Intelligence* 20 (2004) 308-334.
5. P. Ferraris, J. Lee and V. Lifschitz, A generalization of the Lin-Zhao theorem. *Annals of Mathematics and Artificial Intelligence* 47 (2006) 79-101.
6. S. Grell, K. Konczak, and T. Schaub, nomore++ : A system for computing preferred answer sets. In *Proceedings of the Eighth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2005)*, pages 394-398, 2005.
7. F. Lin and Y. Zhao, ASSAT: computing answer sets of a logic program by SAT solvers. *Artificial Intelligence* 157 (2004) 115-137.
8. Y. Zhang, Logic program based updates. *ACM Transactions on Computational Logic* 7 (2006) 421-472.

[3] $Atoms(\Pi)$ plus the relation and inversion 'indicator' atoms respectively.