3. Answer Sets of Propositional Formulas

The definition of answer sets can be extended to arbitrary propositional formulas. We will define the syntax of propositional formulas slightly different from the one in Handout 1, by choosing a different "adequate" set of connectives: $\{\bot, \wedge, \vee, \rightarrow\}$. We use

$$\begin{array}{rll} \top & \text{as shorthand for} & \bot \rightarrow \bot, \\ \neg F & \text{as shorthand for} & F \rightarrow \bot, \\ F \leftrightarrow G & \text{as shorthand for} & (F \rightarrow G) \wedge (G \rightarrow F). \end{array}$$

Recall that atoms and negated atoms are called *literals*.

We use

$$\begin{array}{rll} F, G & \text{as alternative notation for} & F \wedge G, \\ F; G & \text{as alternative notation for} & F \vee G, \\ F \leftarrow G & \text{as alternative notation for} & G \rightarrow F, \\ not\ F \text{ and } \leftarrow F & \text{as alternative notation for} & \neg F. \end{array}$$

The definition of an answer set was extended to more general classes of programs each of which can be viewed as a special class of propositional formulas. These generalizations play an important role in answer set programming. In [Gelfond and Lifschitz, 1991], the head of a rule is allowed to be a disjunction of several atoms, rather than a single atom, for instance:

$$p; q \leftarrow r \tag{1}$$

(in logic programming, disjunction is usually written as a semicolon, just as conjunction is written as a comma). In particular, the disjunction in the head can be empty:

$$\leftarrow r. \tag{2}$$

In a further generalization [Lifschitz and Woo, 1992], the head of a rule may contain negated atoms:

$$p; not\ q \leftarrow r. \tag{3}$$

According to the still more general "nested" syntax of rules [Lifschitz *et al.*, 1999], the head and the body of a rule are arbitrary formulas built from

atoms using any number of conjunctions, disjunctions and negations in any order:

$$(p; not\ q), r \leftarrow not(q, not\ r). \tag{4}$$

Here rules become essentially arbitrary implications that contain no other implications in their antecedents (bodies) and consequents (heads).

Furthermore, the analysis of answer sets in [Pearce, 1997] suggests a simple generalization of the definition of the reduct to arbitrary propositional formulas, including even those that contain nested implications [Ferraris, 2005; Ferraris and Lifschitz, 2005]. The theory of answer sets presented below follows the approach of the last two papers, except that we limit attention to finite programs.

As we understand a traditional rule as an implication whose antecedent is a conjunction of literals, and the consequent is an atom, rules (1)–(4) are identical to the formulas

$$\begin{gathered} r \to (p \vee q), \\ \neg r, \\ r \to (p \vee \neg q), \\ \neg(q \wedge \neg r) \to ((p \vee \neg q) \wedge r). \end{gathered}$$

On the other hand, the formula

$$(p \to q) \to r \tag{5}$$

can be written as a "rule with an embedded implication"

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

or, alternatively,

$$r \leftarrow (q \leftarrow p).$$

From now on, the terms "formula" and "rule" will be used interchangeably. A *program* is a finite set of rules. We will identify a program with the conjunction of its rules. For instance, traditional program (2) in the last handout can be written as the formula

$$p \wedge ((p \wedge q) \to r). \tag{6}$$

Reducts and Answer Sets

The *reduct* F^X of a formula F relative to a set X of atoms is the formula obtained from F by replacing each maximal subformula that is not satisfied

by X with $\bot$. We say that X is an *answer set* of F if X is a minimal set satisfying F^X.

The minimality of X is understood here in the sense of set inclusion. We can verify that X is an answer set of F as follows:

(i) mark in F the maximal subformulas that are not satisfied by X;

(ii) replace each of these subformulas with $\bot$ (after that, equivalent transformations can be used to simplify the result);

(iii) check that the resulting formula is satisfied by X;

(iv) check that it is not satisfied by any proper subset of X.

For instance, to check that $\{p\}$ is an answer set of (6), we do the following:

(i) mark the maximal subformulas of (6) that are not satisfied by $\{p\}$:

$$p \wedge ((\underline{p \wedge q}) \rightarrow \underline{r});$$

(ii) replace these subformulas with $\bot$:

$$p \wedge (\bot \rightarrow \bot);$$

simplify:

$$p;$$

(iii) check that the last formula is satisfied by $\{p\}$;

(iv) check that it is not satisfied by $\emptyset$.

3.1 For each subset of $\{p, q, r\}$ other than $\{p\}$ check that it is not an answer set of (6).

3.2 (a) Formula $\neg\neg p$ has no answer sets. (b) Each of the two formulas

$$p \vee \neg p, \ \neg\neg p \rightarrow p \tag{7}$$

has two answer sets: $\emptyset$ and $\{p\}$.

In application to formulas corresponding to traditional programs, the new definition of the reduct, is, generally, *not* equivalent to the definition from Handout 2.

3.3 Find an example where these two definitions are not equivalent to each other.

However, the new definition of an answer set, when applied to a traditional program written as a formula, turns out to be exactly equivalent to the old definition [Ferraris, 2005]:

Theorem on Traditional Programs. *The answer sets of a traditional program in the sense of Handout 2 are identical to the answer sets of the corresponding formula.*

In particular, a formula corresponding to a positive traditional program, such as (6), has a unique answer set, which is the smallest set of atoms satisfying that formula.

3.4 Verify the assertion of the theorem on traditional programs for the one-rule program

$$p \leftarrow not\ q.$$

(Recall that $\neg q$, according to our convention, is shorthand for $q \to \bot$.)

3.5 Given a formula F and a set X of atoms, let F_X be the formula obtained from F by replacing all occurrences of atoms that do not belong to X by $\bot$. Show that F^X and $(F_X)^X$ are equivalent to each other.

This fact provides an alternative definition of a reduct. A set X of atoms is an answer set of a formula F iff X is the smallest set of atoms satisfying $(F_X)^X$.

It is easy to see that for any X

$$\bot^X = \bot;$$

$$A^X = \begin{cases} A, & \text{if } X \models A, \\ \bot, & \text{otherwise} \end{cases}$$

(A is an atom);

$$(F \odot G)^X = \begin{cases} F^X \odot G^X, & \text{if } X \models F \odot G, \\ \bot, & \text{otherwise} \end{cases}$$

($\odot$ is $\wedge$, $\vee$ or $\to$).

These equalities provide yet another alternative definition of a reduct, which is recursive.

3.6 For any formula F and any set X of atoms,

$$(\neg F)^X = \begin{cases} \bot, & \text{if } X \models F, \\ \top, & \text{otherwise.} \end{cases}$$

Answer Set Semantics

It is easy to see that any answer set of F satisfies F. Indeed, if X doesn't satisfy F then the reduct F^X is $\bot$, so that it cannot be satisfied by X. In this sense, the answer set semantics of propositional formulas is stronger than the classical semantics, which is given by the concept of satisfaction. The sets satisfying a formula are sometimes called its "models," and the answer sets of a formula are called its "stable" models.

The classical semantics of propositional formulas is monotonic, in the sense that conjoining a formula with another formula can only make the set of its models smaller: if X satisfies $F \wedge G$ then X satisfies F. The answer set semantics is nonmonotonic: an answer set of $F \wedge G$ does not have to be an answer set of F. This phenomenon is observed even when F and G are positive traditional programs, and it is related to the minimality condition in the definition of an answer set. For instance, the answer set $\{p, q\}$ of $p \wedge q$ is not an answer set of p. The usefulness of nonmonotonicity as a property of knowledge representation formalisms and its relation to minimality conditions were established in early work on circumscription [McCarthy, 1980].

The classical semantics of propositional formulas and the answer set semantics occupy different places in the polynomial hierarchy: satisfiability is an NP-complete property, and the existence of an answer set is Σ_2^P-complete [Eiter and Gottlob, 1993]. This jump in complexity occurs as soon as the concept of an answer set is extended to disjunctive rules, such as (1); for traditional programs, the existence of an answer set is in class NP.

3.7 (a) Formula $p \vee q$ has two answer sets: $\{p\}$ and $\{q\}$. (b) Formula

$$(p \vee q) \wedge (p \leftrightarrow q)$$

has one answer set: $\{p, q\}$.

These facts provide one more example of nonmonotonicity: after appending the conjunctive term $p \leftrightarrow q$, the formula $p \vee q$ gets a new answer set. They show also that disjunction, under the answer set semantics, is sometimes "exclusive," and sometimes not. On the one hand, among the three sets

$$\{p\},\ \{q\},\ \{p, q\}$$

that satisfy $p \vee q$, the third is not an answer set. On the other hand, appending the $p \leftrightarrow q$ to the disjunction makes $\{p, q\}$ an answer set; the disjunction "becomes inclusive."

Head Atoms

In Handout 2 we discussed two useful properties of answer sets of traditional programs. First, every element of an answer set of a traditional program is the head of one of its rules (Problem 2.3). Second, according to Problem 2.4(b), the answer sets of a traditional program form an "anti-chain": one of them cannot be a proper subset of another.

The examples from Problem 3.2(b) show that the anti-chain property does not extend to arbitrary formulas. As to the assertion of Problem 2.3, it can be extended to formulas as follows.

An occurrence of an atom in a formula F is called positive if the number of implications containing that occurrence in the antecedent is even. An occurrence of an atom in F is *strictly positive* if that number is 0, that is to say, if that occurrence does not belong to the antecedent of any implication in F. Clearly, a formula of the form $\neg F$ has no strictly positive occurrences of atoms (recall that $\neg F$ stands for $F \to \bot$).

We say that A is a *head atom* of F if at least one occurrence of A in F is strictly positive. For instance, the head atoms of a traditional program are the heads of its rules. The only head atom of (3) is p.

Theorem on Head Atoms. *If X is an answer set of a formula F then every element of X is a head atom of F.*

3.8 What does the theorem on head atoms say about the answer sets of formula (5)? Find its answer sets.

References

[Eiter and Gottlob, 1993] Thomas Eiter and Georg Gottlob. Complexity results for disjunctive logic programming and application to nonmonotonic logics. In Dale Miller, editor, *Proceedings of International Logic Programming Symposium (ILPS)*, pages 266–278, 1993.

[Ferraris and Lifschitz, 2005] Paolo Ferraris and Vladimir Lifschitz. Mathematical foundations of answer set programming. In *We Will Show Them! Essays in Honour of Dov Gabbay*, pages 615–664. King's College Publications, 2005.

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

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[Lifschitz and Woo, 1992] Vladimir Lifschitz and Thomas Woo. Answer sets in general nonmonotonic reasoning (preliminary report). In Bernhard Nebel, Charles Rich, and William Swartout, editors, *Proceedings of International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pages 603–614, 1992.

[Lifschitz *et al.*, 1999] Vladimir Lifschitz, Lappoon R. Tang, and Hudson Turner. Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, 25:369–389, 1999.

[McCarthy, 1980] John McCarthy. Circumscription—a form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–39,171–172, 1980.

[Pearce, 1997] David Pearce. A new logical characterization of stable models and answer sets. In Jürgen Dix, Luis Pereira, and Teodor Przymusinski, editors, *Non-Monotonic Extensions of Logic Programming (Lecture Notes in Artificial Intelligence 1216)*, pages 57–70. Springer-Verlag, 1997.