Program Completion

The definitions below are based on [Clark, 1978] and [Lloyd and Topor, 1984]. We begin with a finite signature σ (in the sense of first-order logic) that has no function constants of arity > 0.

A *rule* is a first-order formula of σ that has the form

$$F \to P(\mathbf{t}), \tag{1}$$

where F is a formula, P a predicate constant, and $\mathbf{t}$ a tuple of terms. We will usually write (1) as

$$P(\mathbf{t}) \leftarrow F$$

and call $P(\mathbf{t})$ the *head* of the rule, and F its *body*. We will identify an atomic formula $P(\mathbf{t})$ with the rule $P(\mathbf{t}) \leftarrow \top$.

A *logic program* is a finite set of rules.

The *completion formula* for an n-ary predicate constant P relative to a logic program Π is the sentence obtained as follows:

(i) choose n variables $x_1, \dots, x_n$ that are pairwise distinct and do not occur in Π;

(ii) for each rule

$$P(t_1, \dots, t_n) \leftarrow F$$

of Π containing P in the head, form the rule

$$P(x_1, \dots, x_n) \leftarrow F \wedge x_1 = t_1 \wedge \cdots \wedge x_n = t_n;$$

(iii) for each of the rules

$$P(\mathbf{x}) \leftarrow F$$

obtained on the previous step, make the list $\mathbf{y}$ of all variables that occur in its body F but do not occur in its head $P(\mathbf{x})$, and replace the body F of the rule with $\exists \mathbf{y} F$;

(iv) take all rules

$$P(\mathbf{x}) \leftarrow F_i \qquad (i = 1, \dots, k)$$

obtained on the previous step, and form the sentence

$$\forall \mathbf{x}(P(\mathbf{x}) \leftrightarrow F_1 \vee \cdots \vee F_k).$$

The *completion* of a logic program Π is the set consisting of the completion sentences for all predicate constants of σ relative to Π and the formulas

$$c_i \neq c_j \qquad (1 \leq i < j \leq m),$$

where $c_1, \dots, c_m$ are the object constants of σ. (These formulas are said to express the *unique name assumption*.)

Problem 13. (a) Form the completion of the program consisting of the rules

$$P(a),\ Q(b). \tag{2}$$

(b) Show that the formulas $\neg P(b)$, $\neg Q(a)$ are not entailed by the rules of this program, but are entailed by its completion.

Problem 14. (a) Form the completion of the program consisting of the rules

$$\begin{array}{l} R(x) \leftarrow P(x), \\ R(x) \leftarrow Q(x) \end{array} \tag{3}$$

and simplify it as much as possible. (b) Do the same for the program consisting of rules (2) and (3).

Problem 15. For each of the given logic programs, find all models of its completion:

(a) $P \leftarrow \neg Q$;

(b) $P \leftarrow \neg Q,\ Q \leftarrow \neg R$;

(c) $P \leftarrow \neg Q,\ Q \leftarrow \neg P$.

References

[Clark, 1978] Keith Clark. Negation as failure. In Herve Gallaire and Jack Minker, editors, *Logic and Data Bases*, pages 293–322. Plenum Press, New York, 1978.

[Lloyd and Topor, 1984] John Lloyd and Rodney Topor. Making Prolog more expressive. *Journal of Logic Programming*, 3:225–240, 1984.