TLP **16** (4): 498–508, 2016. © Cambridge University Press 2016
doi:10.1017/S1471068416000089 First published online 3 May 2016

On definite program answers and least Herbrand models

WŁODZIMIERZ DRABENT

Institute of Computer Science, Polish Academy of Sciences,
ul. Jana Kazimierza 5, 01-248 Warszawa, Poland
Department of Computer and Information Science, Linköping University
S – 581 83 Linköping, Sweden
(*e-mail:* drabent@ipipan.waw.pl)

submitted 12 March 2015; revised 16 November 2015; accepted 2 January 2016

Abstract

A sufficient and necessary condition is given under which least Herbrand models exactly characterize the answers of definite clause programs.

KEYWORDS: logic programming, least Herbrand model, declarative semantics, function symbols

1 Introduction

The relation between answers of definite logic programs and their least Herbrand models is not trivial. In some cases, the equivalence

$$\mathscr{M}_P \models Q \quad \text{iff} \quad P \models Q \tag{1}$$

does not hold (where P is a definite program, $\mathscr{M}_P$ its least Herbrand model, and Q a query, i.e. a conjunction of atoms[1]). So programs with the same least Herbrand model may have different sets of answers. (By definition, Q is an answer of P iff $P \models Q$.) For a simple counterexample (Doets 1994, Exercise 4.5), assume that the underlying language has only one function symbol, a constant a. Take a program $P = \{ p(a) \}$. Now $\mathscr{M}_P \models p(X)$ but $P \not\models p(X)$. This counterexample can be in a natural way generalized for any finite set of function symbols, see the comment following the proof of Proposition 15.

Equivalence (1) holds for ground queries (Lloyd 1987, Theorem 6.6; Apt 1997, Theorem 4.30). For a possibly non-ground Q (and a finite P) a sufficient condition for (1) is that there are infinitely many constants in the underlying language (Maher 1988; Apt 1997, Corollary 4.39). Maher (1988) states without proof that instead of

[1] The semantics of non-closed formulae is understood as usually (see e.g. van Dalen (2004), Apt (1997)), so that $IT \models Q$ iff $IT \models \forall Q$, where IT is an interpretation or a theory, Q a formula, and $\forall Q$ its universal closure.

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

an infinite supply of constants it is sufficient that there is a non-constant function symbol not occurring in P, Q. The author is not aware of any proof of this property (except for Drabent (2016, Appendix)).

This paper presents a more general sufficient condition, and shows that the condition is also a necessary one. To obtain the sufficient condition, we show a property of (possibly non-ground) atoms containing symbols not occurring in a program P. Namely, when such atom is true in $\mathscr{M}_P$ then, under certain conditions, a certain more general atom is a logical consequence of P. As an initial step, we obtain a generalization of the theorem on constants (Shoenfield 1967), for a restricted class of theories, namely definite clause programs. We also give an alternative proof for the original theorem.

1.1 Related problem

This paper studies (in)equivalence of two views at the declarative semantics of definite clause programs. One of them considers answers true in the least Herbrand models of programs, the other – answers that are logical consequences of programs.

The subject of this paper should be compared with a related issue (which is outside of the scope of this paper). There exists (in)equivalence between the declarative semantics and the operational one, given by SLD-resolution. As possibly first pointed in Drabent and Maluszynski (1987, 1988), two logically equivalent programs (i.e. with the same models, and thus the same logical consequences) may have different sets of SLD-computed answers for the same query. For instance, take $P_1 = \{\, p(X). \,\}$ and $P_2 = \{\, p(X).\ p(a). \,\}$ Then for a query $p(Y)$ program P_2 gives two distinct computed answers, and P_1 one. This phenomenon gave rise to the *s-semantics*, see e.g. Bossi (2009) for overview and references.

1.2 Preliminaries

We consider definite clause logic programs. A query is a conjunction of atoms. A query Q is an *answer* (or a *correct answer*) of a program P iff $P \models Q$. Apt (1997) calls it a correct instance (of some query). We do not need to refer to SLD-computed answers, as each computed answer is an answer, and each answer is a computed answer for some query, by soundness and completeness of SLD-resolution. Similarly, we do not need to consider to which query Q_0 a given query is an answer.

The Herbrand universe (for the alphabet of function symbols of the underlying language) will be denoted by $\mathscr{HU}$, and the least Herbrand model of a program P by $\mathscr{M}_P$. Remember that $\mathscr{M}_P$ depends on the underlying language. We require $\mathscr{HU} \neq \emptyset$. Names of variables will begin with an upper-case letter. Otherwise, we use the standard definitions and notation of Apt (1997), including the list notation of Prolog. (However, in discussing the semantics of first-order formulae, we use a standard term "variable assignment" instead of "state" used by Apt (1997).)

The paper is organized as follows. The next section presents some necessary definitions. Section 3 shows how existence of answers containing symbols not occurring in the program implies existence of more general answers. The main

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

result of this section is compared with theorem on constants (Shoenfield 1967). Section 4 contains the central technical lemma of this paper. Section 5 studies when the least Herbrand models provide an exact characterization of program answers. A new sufficient condition for equivalence (1) is presented, and it is shown in which sense the condition is a necessary one.

2 Definitions

This section introduces three notions needed further on. Let $\mathcal{F}$ be the set of function symbols of the underlying language; let $F \subseteq \mathcal{F}$. An *alien* w.r.t. F is a non-variable term with its main function symbol from $\mathcal{F} \setminus F$. An alien w.r.t. a theory T (for instance a program) means an alien w.r.t. the set of function symbols occurring in T. An occurrence of an alien t (w.r.t. F, in an atom or substitution) will be called a *maximal alien* if the occurrence is not within an alien $t' \neq t$.

By a generalization of a query, we mean the result of systematic replacement of maximal aliens in the query by new variables. More formally, let $\mathscr{P}$ be a theory or a set of function symbols. Let the maximal aliens of a query Q w.r.t. $\mathscr{P}$ be the occurrences in Q of distinct terms $t_1, \ldots, t_n$. Let $V_1, \ldots, V_n$ be distinct variables not occurring in Q. Let a query Q' be obtained from Q by replacing (each occurrence of) t_i by V_i, for $i = 1, \ldots, n$. (So $Q = Q'\{V_1/t_1, \ldots, V_n/t_n\}$.) Such Q' will be called Q *generalized* for $\mathscr{P}$. We will also call it a/the *generalization* of Q (for $\mathscr{P}$). Note that it is unique up to variable renaming.

Example 1
The standard append program APPEND (Apt 1997, p. 127) contains two function symbols [] and [|]. Terms $a, f([a,b])$ are aliens w.r.t. APPEND, term $[a,b]$ is not. Maximal aliens in $A = app([a],[[]\,|\,g(a,X)],[g(a,Y),Z,[a]])$ are the first and the last occurrences of a and the (single) occurrences of $g(a,X)$ and $g(a,Y)$. Atom $app([V_1],[[]|V_2],[V_3,Z,[V_1]])$ is A generalized for APPEND.

Let Q' be a query not containing aliens w.r.t. $\mathscr{P}$, and θ be a substitution such that $Dom(\theta) \subseteq Var(Q')$. Then Q' is a generalization of $Q'\theta$ for $\mathscr{P}$ (and for $\mathscr{P} \cup \{Q'\}$) iff $\theta = \{V_1/t_1, \ldots, V_n/t_n\}$ where $t_1, \ldots, t_n$ are distinct aliens w.r.t. $\mathscr{P}$.

The correspondence between a ground atom and its generalization is described, in other terms, in Naish (2014, Definition 4). It is used in that paper to represent non-ground atoms by ground ones, in analysis of floundering in the context of delays.

3 On program answers and aliens

Given a query containing aliens which is an answer of a program P, this section shows which more general queries are answers of P. The main result (Lemma 3) is compared with theorem on constants, used by Maher (1988) to prove equivalence (1) for a case with an infinite alphabet of constants.

It is rather obvious that answers containing aliens can be generalized. Assume that a query Q is an answer of P, and that Q contains aliens w.r.t. P. Then Q is

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

a proper instance of some computed answer Q'. It is however not obvious which replacements of aliens in Q by variables result in answers.

Example 2
By replacing aliens w.r.t. P by variables in an answer Q, we obtain some queries which are answers of P, and some which are not. Let $P = \{p(X, X, Y)\}$ and $Q = p(f(a), f(a), b)$. So $P \models Q$. Now $p(f(V_1), V_2, b)$ and $p(V_1, V_2, b)$ are not answers of P, but $p(f(V), f(V), Z)$, $p(V, V, b)$, and $p(V, V, Z)$ are.

Lemma 3
Let P be a program, Q a query, and $\rho = \{V_1/t_1, \ldots, V_k/t_k\}$ be a substitution where $t_1, \ldots, t_k$ are distinct aliens w.r.t. $P \cup \{Q\}$. Then

$$P \models Q \quad \text{iff} \quad P \models Q\rho\,. \tag{2}$$

Note that terms $t_1, \ldots, t_k$ may be non-ground (and may contain variables from $\{V_1, \ldots, V_k\}$), some t_i, t_j may be unifiable, or contain common variables, Q may contain variables other than $V_1, \ldots, V_n$ and may contain aliens w.r.t. P. So Q is not necessarily a generalization of $Q\rho$ for P, but it is one for $P \cup \{Q\}$.

Example 4
In the previous example, the cases in which the more general atom is an answer of P satisfy conditions of Lemma 3, and the remaining ones do not.

Proof (Lemma 3)
Without loss of generality assume that variables $V_1, \ldots, V_k$ occur in Q. Let $X_1, \ldots, X_l$ be the remaining variables of Q. The "only if" case is obvious.

Assume $P \models Q\rho$. By completeness of SLD-resolution, $Q\rho$ is an instance of some computed answer $Q\varphi$ for P and Q: $Q\rho = Q\varphi\sigma$. Each function symbol occurring in φ occurs in P or Q. Moreover (for $i = 1, \ldots, k$), $t_i = V_i\varphi\sigma$ and the main symbol of t_i does not occur in $V_i\varphi$; hence $V_i\varphi$ is a variable. As $t_1, \ldots, t_k$ are distinct, variables $V_1\varphi, \ldots, V_k\varphi$ are distinct. Similarly, $X_j = X_j\varphi\sigma$ for $j = 1, \ldots, l$, thus $V_1\varphi, \ldots, V_k\varphi, X_1\varphi, \ldots, X_l\varphi$ are distinct variables. Thus $Q\varphi$ is a variant of Q and, by soundness of SLD-resolution, $P \models Q$. □

Corollary 5
Let P be a program, Q a query, and Q' be Q generalized for P. Then $P \models Q$ iff $P \models Q'$.

Proof
$Q = Q'\rho$ for a certain $\rho = \{V_1/t_1, \ldots, V_k/t_k\}$. The premises of Lemma 3 are satisfied by P, Q', and ρ (as $t_1, \ldots, t_k$ are aliens w.r.t. P, but also w.r.t. Q'). □

Example 6
Consider again program APPEND. Assume that the underlying language has more function symbols than those occurring in the program, i.e. [], [|]. Assume that we know that the least Herbrand model $\mathscr{M}_{\text{APPEND}}$ contains an atom $Q = app([t_1, \ldots, t_m], [t_{m+1}, \ldots, t_k], [t_1, \ldots, t_k])$, where $t_1, \ldots, t_k$ are distinct aliens w.r.t. APPEND. Note that $P \models Q$, as equivalence (1) holds for ground queries.

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

By Corollary 5, APPEND $\models app([V_1,\ldots,V_m],[V_{m+1},\ldots,V_k],[V_1,\ldots,V_k])$, where $V_1,\ldots,V_k$ are distinct variables. Hence, for any terms $s_1,\ldots,s_k$, APPEND $\models app([s_1,\ldots,s_m],[s_{m+1},\ldots,s_k],[s_1,\ldots,s_k])$.

Example 7
Consider the map colouring program (Sterling and Shapiro 1994, Program 14.4). We skip any details, let us only mention that the names of colours and countries do not occur in the program. (The function symbols occurring in the program are $F = \{[],[\,|\,],region\}$.) By Corollary 5, for any answer Q of the program, the generalization Q' of Q w.r.t. F is an answer of the program. So is each instance of Q'. Thus systematic replacing (some) names of colours or countries in Q by other terms results in a query Q'' which is an answer of the program.[2]

The proof of equivalence (1) for an infinite set of constants of Maher (1988, proof of Proposition 6) employs a so called theorem on constants (Shoenfield 1967), see also free constant theorem in Davis (1993, p. 56). The theorem states that (2) holds for an arbitrary theory P and formula Q, when the distinct aliens $t_1,\ldots,t_k$ are constants. Its proofs in Shoenfield (1967) and Davis (1993) are syntactical, but a rather simple semantic proof is possible:

Let F be the set of function and predicate symbols from P,Q, let $\mathscr{X}$ be the set of the free variables of Q. Notice that for any interpretation I (for F) and any variable assignment σ (for $\mathscr{X}$) there exists a variable assignment σ' (for $\mathscr{X}\setminus\{V_1,\ldots,V_k\}$) and an interpretation I' (for $F\cup\{t_1,\ldots,t_k\}$) such that $\sigma'(X)=\sigma(X)$ for each $X\in\mathscr{X}\setminus\{V_1,\ldots,V_k\}$, $I'(t_i)=\sigma(V_i)$ for each i, and all the symbols of F have the same interpretation in I and I'. Thus $I\models P$ iff $I'\models P$, and $I\models_\sigma Q$ iff $I'\models_{\sigma'} Q\rho$. Conversely, for each interpretation I' for $F\cup\{t_1,\ldots,t_k\}$ and variable assignment σ' for $\mathscr{X}\setminus\{V_1,\ldots,V_k\}$ there exist I,σ as above. (In particular, the two equivalences hold.) Now the theorem follows:

$P\models Q$ iff
for every I,σ (as above) $I\models P$ implies $I\models_\sigma Q$ iff
for every I',σ' (as above) $I'\models P$ implies $I'\models_{\sigma'} Q\rho$ iff
$P\models Q\rho$.

Maher (1988, p. 634) states that "The same effect [as adding new constants] could be obtained with one new function symbol (of arity >0) to obtain new ground terms with new outermost function symbol." This idea does not apply to the proof of the previous paragraph; when $t_1,\ldots,t_k$ are such terms then the proof fails.[3] So do the proofs of Shoenfield (1967) and Davis (1993). In the context of Shoenfield (1967) – first-order logic with equality – the generalization of the theorem on constants to terms with a new outermost symbol does not hold. For a counterexample, note that

[2] Thus it is possible that neighbouring countries get the same colour. This does not mean that the program is incorrect. Its main predicate *color_map* describes a correct map colouring provided that its second argument is a list of distinct colours.

[3] Informally, this is because such new terms cannot be interpreted independently, in contrast to k new constants. Sometimes no interpretation for the new symbol f is possible, such that $t_1,\ldots,t_k$ are interpreted as a given k values. For instance, take $t_i=f^i(a)$ for $i=1,\ldots,k$. Then for any interpretation for f, if t_1,t_2 have the same value then all $t_1,\ldots,t_k$ also have the same value.

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

$\{a = b\} \models f(a) = f(b)$ but $\{a = b\} \not\models V_1 = V_2$. The generalization in Lemma 3 is sound and has a simple proof, due to restriction to definite programs and queries.

From Lemma 3 it follows that equivalence (1) holds whenever the underlying language has a non-constant function symbol f (or a sufficient number of constants) not occurring in P, Q.[4] (See also Drabent (2016, Appendix) for a direct proof.) We however aim for a more general sufficient condition for (1), allowing f to occur in Q; in this case Lemma 3 is not applicable.

4 Least Herbrand models and program answers

This section shows conditions under which truth in $\mathscr{M}_P$ of a query with aliens implies that a certain more general query is an answer of P. This is a central technical result of this paper (Lemma 10). From it, the sufficient conditions for equivalence (1) follow rather straightforwardly, as shown in the next section. We begin with proving an auxiliary property, by means the two following lemmas:

Lemma 8
Two distinct terms have at most one unifier of the form $\{X/u\}$ where u is not a variable.

Proof
Let $\theta = \{X/u\}$, $\theta' = \{X'/u'\}$ be distinct substitutions, where neither of u, u' is a variable. We show that if $s_1\theta = s_2\theta$ then $s_1\theta' \neq s_2\theta'$, for any distinct terms s_1, s_2. The proof is by induction on the sum $|s_1| + |s_2|$ of the sizes of s_1, s_2. (Any notion of term size would do, providing that $|t| < |t'|$ whenever t is a proper subterm of t'.) Assume that the property holds for each s'_1, s'_2 such that $|s'_1| + |s'_2| < |s_1| + |s_2|$.

Let $s_1 \neq s_2$ and $s_1\theta = s_2\theta$. Notice that at most one of s_1, s_2 is a variable. (Otherwise $s_1\theta, s_2\theta$ are s_1, s_2 – two distinct variables, or exactly one of $s_1\theta, s_2\theta$ is a variable, contradiction.) Assume that exactly one of s_1, s_2, say s_1, is a variable. Then $s_1 = X$ (as $s_1\theta \neq s_1$), so X does not occur in s_2 (as X, s_2 are unifiable), hence $s_2\theta = s_2 = u$. Now if $X' \neq X$ then $s_1\theta' = X$ which is distinct from any instance of s_2. Otherwise $X' = X$, hence $s_1\theta' = u' \neq u = s_2 = s_2\theta'$.

If both s_1, s_2 are not variables then $s_i = f(s_{i1}, \ldots, s_{il})$, for $i = 1, 2$. For some j, $s_{1j} \neq s_{2j}$ and $|s_{1j}| + |s_{2j}| < |s_1| + |s_2|$. By the inductive assumption, $s_{1j}\theta' \neq s_{2j}\theta'$; thus $s_1\theta' \neq s_2\theta'$. □

Lemma 9
let $\mathscr{P}$ be a theory or a set of function symbols. Let $t_1, \ldots, t_m$ be a sequence of distinct terms, where $t_1, \ldots, t_n$ $(0 \leqslant n \leqslant m)$ are variables, and $t_{n+1}, \ldots, t_m$ are aliens w.r.t. $\mathscr{P}$. Assume that if $t_{n+1}, \ldots, t_m$ are ground then there exist ground aliens $u_1, \ldots, u_n$ w.r.t. $\mathscr{P}$, pairwise distinct from $t_{n+1}, \ldots, t_m$. Then the sequence has a ground instance $(t_1, \ldots, t_m)\sigma$ consisting of m distinct aliens w.r.t. $\mathscr{P}$.

[4] Assume that $V_1, \ldots, V_k$ are the variables of Q, and that there exist distinct ground terms $t_1, \ldots, t_k$ with their main symbols not occurring in P, Q. Let $\rho = \{V_1/t_1, \ldots, V_k/t_k\}$. Assume $\mathscr{M}_P \models Q$, so $\mathscr{M}_P \models Q\rho$, and $P \models Q\rho$ as $Q\rho$ is ground. By Lemma 3, $P \models Q$.

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

Proof

Consider first the case of $t_{n+1},\ldots,t_m$ ground. Then $\sigma = \{t_1/u_1,\ldots,t_n/u_n\}$ is a substitution providing the required instance.

Let some t_j $(n < j \leqslant m)$ be non-ground. Its main symbol, say f, is a non-constant function symbol not occurring in $\mathscr{P}$. Thus the set Al of ground aliens w.r.t. $\mathscr{P}$ is infinite.

Let $X_1,\ldots,X_l$ be the variables occurring in $t_1,\ldots,t_m$. For some $s_1 \in Al$, substitution $\theta_1 = \{X_1/s_1\}$ is not a unifier of any pair t_i, t_j $(1 \leqslant i < j \leqslant m)$, as by Lemma 8 each such pair has at most one unifier of the form $\{X_1/s\}$, $s \in \mathscr{HU}$. Thus $(t_1,\ldots,t_m)\theta_1$ is a sequence of m distinct terms. Applying this step repetitively, we obtain the required sequence $(t_1,\ldots,t_m)\theta_1 \cdots \theta_l$ of distinct ground terms. □

Lemma 10

Let P be a program, Q an atom, and Q' be Q generalized for P. If

1. the underlying language has a non-constant function symbol not occurring in P, or
2. Q contains exactly $n \geqslant 0$ (distinct) variables, and the underlying language has (at least) n constants not occurring in P, Q,

then $\mathscr{M}_P \models Q$ iff $P \models Q'$.

Proof

Note that $Q = Q'\varphi$ where $\varphi = \{ X_1/u_1,\ldots X_m/u_m \}$, $X_1,\ldots,X_m$ are the variables of Q' not occurring in Q, and $u_1,\ldots,u_m$ are the maximal aliens in Q (precisely: the distinct terms whose occurrences in Q are the maximal aliens w.r.t. P). Let $Y_1,\ldots,Y_n$ be the variables occurring in Q.

We construct a ground instance $Q\sigma$ of Q, such that Q' is $Q\sigma$ generalized for P. To apply Lemma 9 to terms $Y_1,\ldots,Y_n,u_1,\ldots,u_m$, note that if $u_1,\ldots,u_m$ are ground then there exist n ground aliens w.r.t. P pairwise distinct from $u_1,\ldots,u_m$. (They are either the constants from condition 2, or can be taken from the infinite set of ground aliens w.r.t. P with the main symbol from condition 1.) By Lemma 9, there exists a ground instance $(Y_1,\ldots,Y_n,u_1,\ldots,u_m)\sigma$, consisting of $n+m$ distinct aliens w.r.t. P, where the domain of σ is $\{Y_1,\ldots,Y_n\}$.

Note that $\varphi\sigma = \sigma \cup \{ X_1/u_1\sigma,\ldots,X_m/u_m\sigma \}$. The substitution maps variables $Y_1,\ldots,Y_n,X_1,\ldots,X_m$ to distinct aliens $(Y_1,\ldots,Y_n,u_1,\ldots,u_m)\sigma$ w.r.t. P. So Q' is $Q'\varphi\sigma$ generalized for P. Thus $P, Q'\varphi\sigma$, and Q' satisfy the conditions of Corollary 5.

Now $\mathscr{M}_P \models Q$ implies $\mathscr{M}_P \models Q\sigma$ and then $P \models Q\sigma$ (as equivalence (1) from Introduction holds for ground queries). As $Q\sigma = Q'\varphi\sigma$, by Corollary 5 $P \models Q'$. The "if" case is obvious, as Q is an instance of Q'. □

Remark 11

The premises of Lemma 10 can be weakened by stating that Q, Q' are atoms such that $Q = Q'\varphi$ for a substitution $\varphi = \{ X_1/u_1,\ldots X_m/u_m \}$, where $u_1,\ldots,u_m$ are distinct aliens w.r.t. $P \cup \{Q'\}$, and variables $X_1,\ldots,X_m$ do not occur in Q.

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

Proof
Obtained by minor modifications of the proof above. The first sentence, describing φ, is to be dropped. Each "w.r.t. P" is to be changed to "w.r.t. $P \cup \{Q'\}$". In the third paragraph, substitution $\varphi\sigma$ together with P and Q' satisfy the condition of Lemma 3. At the end of the proof, Lemma 3 should be applied instead of Corollary 5. □

It remains to generalize Lemma 10 to arbitrary queries.

Corollary 12
Lemma 10 also holds for non-atomic queries. Moreover, condition 2 of the lemma can be replaced by the following:

3. for each atom A of Q with $k \geqslant 0$ (distinct) variables, the underlying language has (at least) k constants not occurring in P, A.

Proof
Note that condition 2 implies condition 3. So assume that the latter holds. Let $Q = A_1, \ldots, A_l$ generalized for P be $Q' = A'_1, \ldots, A'_l$. Then each A'_i is A_i generalized for P. So Lemma 10 applies to each A_i, A'_i. Thus $\mathscr{M}_P \models Q$ implies $P \models A'_i$, for each $i = 1, \ldots, l$. Hence, $P \models Q'$. □

5 Characterization of program answers by the least Herbrand model

This section studies when the least Herbrand models exactly characterize the program answers. First, a sufficient condition is presented for equivalence (1) from Introduction. Then we show that the sufficient condition is also necessary. Conditions 1, 2 below are the same as conditions 1, 3 of Lemma 10 and Corollary 12.

Theorem 13 (*Characterizing answers by* $\mathscr{M}_P$)
Let P be a program, and Q a query such that

1. the underlying language has a non-constant function symbol not occurring in P, or
2. for each atom A of Q with $k \geqslant 0$ (distinct) variables, the underlying language has (at least) k constants not occurring in P, A.

Then, $\mathscr{M}_P \models Q$ iff $P \models Q$.

Note that condition 1 implies that the equivalence holds for every query Q, including queries containing the new symbol. Also, it holds for every query Q and every finite program P when the alphabet contains infinitely many function symbols, as then condition 1 or 2 is satisfied. From the theorem the known sufficient conditions follow: the alphabet containing infinitely many constants (and P finite), or Q ground.

Condition 2 is implied by its simpler version: the language has $k \geqslant 0$ constants not occurring in P, Q, and each atom of Q contains no more than k variables.

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

Proof of Theorem 13
Let Q' be Q generalized for P. By Corollary 12, $\mathscr{M}_P \models Q$ implies $P \models Q'$, hence $P \models Q$, as Q is an instance of Q'. The reverse implication is obvious. □

We conclude with showing in which sense the sufficient condition of Theorem 13 is also necessary. As expected, it is strictly speaking not a necessary condition for (1), as it is violated for some P, Q for which (1) holds.

Example 14
Consider program APPEND and assume that the only function symbols of the underlying language are [], [|]. Let $Q = app([X], [Y], [X, Y])$. Then $\mathscr{M}_{\mathrm{APPEND}} \models Q$ and APPEND $\models Q$, but the condition of Theorem 13 is violated.

On the other hand, consider a program P of three clauses $app(\,[\,], L, L\,).$; $app(\,[[\,]|K], L, [[\,]|M]\,) \leftarrow app(\,K, L, M\,).$; $app(\,[[H|T]|K], L, [[H|T]|M]\,) \leftarrow app(\,K, L, M\,).$ Programs APPEND and P have the same least Herbrand model but different sets of answers, as e.g. $P \not\models Q$. The condition of Theorem 13 is violated by P, Q, and the equivalence does not hold. Note that P cannot be used to append lists when new function symbols are added to the language; $app([a], [b], [a, b])$ is then not an answer of P.

Roughly speaking, the sufficient conditions of Theorem 13 and Lemma 10 are also necessary, when all what is known about a program is the set of function symbols employed in it. Formally:

Proposition 15
Let $\mathcal{F}$ be the set of function symbols of the underlying language, and $F_0 \subseteq \mathcal{F}$ be its finite subset. Let Q be a query, such that the predicate symbols of the atoms of Q are distinct. Assume that $\mathscr{M}_P \models Q$ iff $P \models Q$, for each finite program P such that F_0 is the set of function symbols occurring in P. Then the sufficient condition of Theorem 13 holds.

The proposition also holds when F_0 and the considered program P are infinite.

Proof
Let Q be a query whose atoms have distinct predicate symbols. Assume that the sufficient condition of Theorem 13 does not hold. We show that for a certain program P (such that F_0 is the set of the function symbols occurring in P), $\mathscr{M}_P \models Q$ but $P \not\models Q$.

As condition 1 of Theorem 13 does not hold, all the non-constant function symbols of $\mathcal{F}$ are in F_0. As condition 2 does not hold, there is an atom A in Q with k distinct variables $Y_1, \ldots, Y_k$, for which the number of constants from $\mathcal{F} \setminus F_0$ not occurring in A is $l < k$; let $a_1, \ldots, a_l$ be the constants. The atom can be represented as $A = B[b_1, \ldots, b_n, Y_1, \ldots, Y_k]$, where $b_1, \ldots, b_n$ are those (distinct) constants of A which are not in F_0.[5] So $\mathcal{F} \setminus F_0 = \{a_1, \ldots, a_l, b_1, \ldots, b_n\}$.

[5] Formally, $B[t_1, \ldots, t_{n+k}]$ can be defined as the instance $B\{V_1/t_1, \ldots V_{n+k}/t_{n+k}\}$ of an atom B, whose (distinct) variables are $V_1, \ldots, V_{n+k}$, and whose function symbols are from F_0.

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

Let P_0 be the set of atoms of Q except for A. Let $\mathscr{V} = \{X_1, \ldots, X_{n+k-1}\}$ be $n+k-1$ distinct variables. Let P consist of the unary clauses of P_0 and the unary clauses of the form $B[t_1, \ldots, t_{n+k}]$ where (i) $\{t_1, \ldots, t_{n+k}\} = \mathscr{V}$ (so a variable occurs twice), or (ii) $\{t_1, \ldots, t_{n+k}\} = \mathscr{V} \cup \{f(\vec{Z})\}$ where $f \in F_0$, its arity is $m \geqslant 0$, and $\vec{Z}$ is a tuple of m distinct variables pairwise distinct from those in $\mathscr{V}$. Note that P is finite iff F_0 is.

Each ground atom $B' = B[u_1, \ldots, u_{n+k}]$ (where $u_1, \ldots, u_{n+k} \in \mathscr{HU}$) is an instance of some clause of P, as if $u_1, \ldots, u_{n+k}$ are distinct terms then the main symbol of some of them is in F_0, and B' is an instance of a clause of the form (ii), otherwise B is an instance of a clause of the form (i). Thus $\mathscr{M}_P \models A$, hence $\mathscr{M}_P \models Q$ (as $P_0 \models A'$ for each atom A' of Q distinct from A). To show that $P \not\models Q$, add new constants $a_{l+1}, \ldots, a_k$ to the alphabet. Then $B[b_1, \ldots, b_n, a_1, \ldots, a_k]$ is not an instance of any clause of P, so $B[b_1, \ldots, b_n, a_1, \ldots, a_k]$ is false in the least Herbrand model of P with the extended alphabet. □

The proof provides a family of counterexamples for a claim that $\mathscr{M}_P \models Q$ and $P \models Q$ are equivalent. In particular, setting $Q = p(V)$ ($k = 1$, $n = 0$) results in $P = \{ p(f(\vec{Z})) \mid f \in F \}$, a generalization of the counterexample from Introduction to any underlying finite set F of function symbols.

From the proposition it follows that a more general sufficient condition (than that of Theorem 13) for the equivalence of $\mathscr{M}_P \models Q$ and $P \models Q$ is impossible, unless it uses more information about P than just the set of involved symbols.

6 Conclusion

In some cases, the least Herbrand model does not characterize the set of answers of a definite program. This paper generalizes the sufficient condition for $\mathscr{M}_P \models Q$ iff $P \models Q$, to "a non-constant function symbol not in P, or k constants not in P, A for each atom A of Q". It also shows that the sufficient condition cannot be improved unless more is known about the program than just which function symbols occur in it. As a side effect, it is shown which more general queries are implied to be answers of P by Q being an answer.

Acknowledgement

Comments of anonymous referees helped improving the presentation.

References

Apt, K. R. 1997. *From Logic Programming to Prolog*. International Series in Computer Science, Prentice-Hall, Hemel Hempstead, Hertfordshire.

Bossi, A. 2009. S-semantics for logic programming: A retrospective look. *Theoretical Computer Science 410,* 46, 4692–4703.

Davis, M. 1993. First order logic. In *Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. 1, Logic Foundations*, D. M. Gabbay, C. J. Hogger and J. A. Robinson, Eds. Oxford University Press, New York, 31–65.

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press

Doets, K. 1994. *From Logic to Logic Programming*. The MIT Press, Cambridge, MA.

Drabent, W. 2016. Correctness and Completeness of Logic Programs. *ACM Trans. Comput. Log. 17*, 3 doi: http://dx.doi.org/10.1145/2898434.

Drabent, W. and Maluszynski, J. 1987. Inductive assertion method for logic programs. In *TAPSOFT'87 (International Joint Conference on Theory and Practice of Software Development, Pisa, Italy), Vol. 2*, H. Ehrig, R. A. Kowalski, G. Levi and U. Montanari, Eds. Lecture Notes in Computer Science, Vol. 250. Springer-Verlag, Berlin, Heidelberg, 167–181.

Drabent, W. and Małuszyński, J. 1988. Inductive assertion method for logic programs. *Theoretical Computer Science 59*, 133–155.

Lloyd, J. W. 1987. *Foundations of Logic Programming*. Springer-Verlag, Berlin, Heidelberg. Second, extended edition.

Maher, M. J. 1988. Equivalences of logic programs. In *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed. Morgan Kaufmann Publishers, Inc. Los Altos California, 627–658.

Naish, L. 2014. Transforming floundering into success. *TPLP 14,* 2, 215–238.

Shoenfield, J. R. 1967. *Mathematical Logic*. Addison-Wesley, Reading, MA.

Sterling, L. and Shapiro, E. 1994. *The Art of Prolog*, 2 ed. The MIT Press, Cambridge, MA.

van Dalen, D. 2004. *Logic and Structure*, 4th ed. Springer-Verlag, Berlin, Heidelberg.

https://doi.org/10.1017/S1471068416000089 Published online by Cambridge University Press