Lógica Computacional
Nota 13. Teoría de Herbrand*

Noé Salomón Hernández S.

1. Interpretación de cláusulas

Una cláusula A : -$B_1, \ldots, B_n$ de PROLOG tiene una interpretación declarativa y una interpretación operacional.

Declarativa A es válida si B_1 y ... y B_n son válidas. Esta interpretación permite discutir la correctud de la cláusula.

Operacional Para ejecutar A basta ejecutar B_1 y ... y B_n. Esta interpretación permite considerar a la cláusula como la definición de un proceso.

En esta nota determinaremos que ambas interpretaciones son equivalentes, para lo cual necesitamos de los siguientes dos conceptos.

Definición 1.1. Sea $\mathbb{P}$ un programa y G una cláusula meta. Una sustitución θ para las variables en G es una *sustitución de respuesta correcta* si $\mathbb{P} \models \forall(\neg G\theta)$, donde la cuantificación universal se toma sobre todas las variables libres en $\neg G\theta$.

Una sustitución de respuesta correcta para $\mathbb{P}$ y G corresponde a una consecuencia lógica particular del programa, y es entonces un significado declarativo del programa.

Definición 1.2. Sea $\mathbb{P}$ un programa y G una cláusula meta. Una sustitución σ es una *sustitución de respuesta computada* para $\mathbb{P} \cup G$ syss existe una refutación SLD de longitud n con unificadores más generales $\mu_0, \ldots, \mu_{n-1}$ tales que $\sigma = \mu_0 \cdots \mu_{n-1}|_{Var(G)}$

Es decir, σ es una sustitución de respuesta computada para $\mathbb{P} \cup G$ syss σ es la restricción de la composición de los unificadores de una refutación SLD para $\mathbb{P} \cup G$. Una respuesta computada corresponde al resultado del proceso de inferencia por parte del sistema. Las sustituciones de respuesta computada son aquellas que el intérprete de PROLOG devuelve al usuario.

Así que estudiaremos la relación entre la sustitución de respuesta correcta y la sustitución de respuesta computada. Vemos que el significado de un programa lógico se debe dar mediante el conjunto de sustituciones, ya sean sustituciones de respuesta correcta o respuesta computada. Dado un programa lógico $\mathbb{P}$, podemos asignarle dos significados:

Declarativo: el significado de un programa lógico $\mathbb{P}$ es el conjunto de todas las consecuencias lógicas del programa, noción asociada a las sustituciones de respuesta correcta.

*Esta nota se base en material elaborado por el prof. Favio Miranda.

Operacional: el significado de un programa lógico $\mathbb{P}$ es el conjunto de todas las refutaciones SLD a partir del programa, noción asociada a las sustituciones de respuesta computada.

En concreto, estudiaremos conceptos y resultados que nos permitan concluir que de *toda consecuencia lógica atómica de* $\mathbb{P}$ podemos obtener una refutación SLD, y viceversa. Es decir, veremos que el método de resolución es completo y correcto para el caso de fórmulas atómicas.

2. Universo y Modelos de Herbrand

Definición 2.1. Sea $\mathcal{L}$ un lenguaje de primer orden. El *universo de Herbrand* de $\mathcal{L}$ se define como el conjunto $\mathcal{H}_\mathcal{L}$ de los términos cerrados de $\mathcal{L}$, es decir,

$$\mathcal{H}_\mathcal{L} = \{t \,|\, t \in \mathcal{L} \text{ y } t \text{ es un término cerrado}\}$$

Si en $\mathcal{L}$ no hubiera constantes, se agrega una para poder formar el universo de Herbrand.

Definición 2.2. Sea $\mathcal{L}$ un lenguaje y $\mathcal{H}_\mathcal{L}$ su universo de Herbrand. La *base de Herbrand* $\mathcal{B}_\mathcal{L}$ de $\mathcal{L}$ se define como el conjunto de fórmulas atómicas cerradas, es decir, el conjunto de fórmulas atómicas cuyos argumentos pertenecen al universo de Herbrand de $\mathcal{L}$.

Definición 2.3. Dado un programa lógico $\mathbb{P}$ definimos el universo y la base de Herbrand, denotados como $\mathcal{H}_\mathbb{P}$ y $\mathcal{B}_\mathbb{P}$ respectivamente, como:

$$\mathcal{H}_\mathbb{P} =_{\mathsf{def}} \mathcal{H}_{\mathcal{L}(\mathbb{P})} \qquad\qquad \mathcal{B}_\mathbb{P} =_{\mathsf{def}} \mathcal{B}_{\mathcal{L}(\mathbb{P})}$$

Definición 2.4. Decimos que un modelo $\mathcal{M}$ de un lenguaje $\mathcal{L}$ sobre el universo $\mathcal{H}_\mathcal{L}$ es un *modelo de Herbrand* syss cumple lo siguiente:

- Para todo símbolo de constante c, se cumple $c^\mathcal{M} = c$.
- Para todo símbolo de función f de aridad n, se cumple $f^\mathcal{M}(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$
- Para todo símbolo de predicado P de aridad n, se tiene que $P^\mathcal{M} \subseteq \mathcal{H}^n_\mathcal{L}$

En un modelo de Herbrand los símbolos de constante y de función se interpretan como ellos mismos y lo único que no está determinado es la interpretación de los símbolos de predicado. De modo que a los modelos de Herbrand se les llama también modelos sintácticos.

2.1. Representación de los modelos de Herbrand

Una manera útil de representar modelos de Herbrand es mediante subconjuntos de la base de Herbrand. Dado un modelo de Herbrand $\mathcal{M}$, representamos a $\mathcal{M}$ mediante el subconjunto $B_\mathcal{M} \subseteq \mathcal{B}_\mathcal{L}$ de aquellas fórmulas atómicas que son verdaderas en $\mathcal{M}$. Como cada subconjunto $B \subseteq \mathcal{B}_\mathcal{L}$ determina de manera única a un modelo de Herbrand $\mathcal{M}$, identificamos a $\mathcal{M}$ con B. Para verificar si una fórmula atómica A es verdadera en $\mathcal{M}$ basta ver que $A \in \mathcal{M}$.

Lema 2.5. *Sea $\mathcal{L}$ un lenguaje, $\mathcal{M}$ un modelo de Herbrand para $\mathcal{L}$, t un término con variables $x_1, \ldots, x_n$ y ℓ un ambiente para el universo $\mathcal{H}_\mathcal{L}$ tal que $\ell(x_i) = r_i$ con $r_i \in \mathcal{H}_\mathcal{L}$ para $1 \leq i \leq n$, es decir, cada r_i es un término cerrado. Entonces,*

$$t^{\mathcal{M}} = t[x_1 := r_1, \ldots, x_n := r_n]$$

En particular, si t es un término cerrado, entonces $t^{\mathcal{M}} = t$.

La interpretación de términos en un modelo de Herbrand coincide con aplicar una sustitución al término. Vemos que los términos cerrados se interpretan como ellos mismos.

3. El Teorema de Herbrand

El teorema de Herbrand es esencial para describir la semántica de los programas lógicos.

Definición 3.1. Sean $\mathcal{L}$ un lenguaje y $\mathcal{K}$ un conjunto de fórmulas en $\mathcal{L}$ cerradas y universales. El conjunto de instancias cerradas $\varphi\sigma$, con $\forall x_1 \ldots \forall x_n \, \varphi \in \mathcal{K}$, se llama el *conjunto de instancias cerradas de* $\mathcal{K}$ y se denota $IC(\mathcal{K})$. Es decir, $IC(\mathcal{K})$ es un conjunto donde aparecen todas las fórmulas φ para toda instancia de $x_1, \ldots, x_n$.

Teorema 3.2 (Teorema de Herbrand)**.** *Sean $\mathcal{L}$ un lenguaje con al menos una constante y $\mathcal{K}$ un conjunto de enunciados universales en $\mathcal{L}$. Las siguientes condiciones son equivalentes:*

a) $\mathcal{K}$ tiene un modelo.

b) $\mathcal{K}$ tiene un modelo de Herbrand.

c) $IC(\mathcal{K})$ tiene un modelo.

d) $IC(\mathcal{K})$ tiene un modelo de Herbrand.

Demostración
Como todo modelo de Herbrand es un modelo, las implicaciones $b \Rightarrow a$ y $d \Rightarrow c$ son triviales. La validez universal de la fórmula $\forall x_1 \ldots \forall x_n \, \varphi \rightarrow \varphi[x_1 := t_1, \ldots, x_n := t_n]$ hace ciertas a las implicaciones $a \Rightarrow c$ y $b \Rightarrow d$

Demostremos la implicación $c \Rightarrow b$. Sea $\mathcal{M}$ un modelo de $IC(\mathcal{K})$. Definimos una estrucutra de Herbrand $\mathcal{N}$ sobre el universo de Herbrand $\mathcal{H}_\mathcal{L}$, por lo que es suficiente indicar la interpretación de los símbolos de predicado, ya que los símbolos de constante y de función se interpretan a sí mismos. Si P es un símbolo de predicado n-ario, definimos

$$P^{\mathcal{N}} = \{(t_1, \ldots, t_n) \,|\, \mathcal{M} \models P(t_1, \ldots, t_n)\}$$

Por construcción, $\mathcal{N}$ cumple que:

$$\mathcal{M} \models P(t_1, \ldots, t_n) \text{ si y solo si } \mathcal{N} \models P(t_1, \ldots, t_n)$$

es decir, $\mathcal{M}$ y $\mathcal{N}$ validan las mismas fórmulas atómicas cerradas. Tal resultado se puede extender a cualquier fórmula cerrada libre de cuantificadores mediante inducción sobre fórmulas[1]. Finalmente, veamos que $\mathcal{N}$ es modelo de $\mathcal{K}$. Sea $\forall x_1 \ldots \forall x_n \, \varphi$ una fórmula de $\mathcal{K}$. Hay que demostrar

[1] **Lema 3.3.** *Si φ es un enunciado sin cuantificadores, entonces $\mathcal{M} \models \varphi$ syss $\mathcal{N} \models \varphi$.*

que $\mathcal{N} \models \forall x_1 \ldots \forall x_n \varphi$, lo cual es equivalente a demostrar que para cualesquiera $t_1, \ldots, t_n \in \mathcal{H}_\mathcal{L}$, $\mathcal{N} \models \varphi[x_1 := t_1, \ldots, x_n := t_n]$. Esto último, por el Lema 3.3 es equivalente a que para cualesquiera $t_1, \ldots, t_n \in \mathcal{H}_\mathcal{L}$, $\mathcal{M} \models \varphi[x_1 := t_1, \ldots, x_n := t_n]$, lo cual es cierto puesto que $\varphi[x_1 := t_1, \ldots, x_n := t_n]$ pertence a $IC(\mathcal{K})$ y, por hipótesis, $\mathcal{M}$ es modelo de $IC(\mathcal{K})$. De manera que $\mathcal{N} \models \varphi[x_1 := t_1, \ldots, x_n := t_n]$. Así, $\mathcal{N}$ es un modelo de Herbrand de $\mathcal{K}$. ■

Dado que las cláusulas son enunciados universales, se tiene el siguiente resultado.

Corolario 3.4. *Sea $\mathcal{C}$ un conjunto de cláusulas de Horn. Las siguientes condiciones son equivalentes:*

a) $\mathcal{C}$ tiene un modelo.

b) $\mathcal{C}$ tiene un modelo de Herbrand.

3.1. Semántica declarativa de los programas lógicos

Debido a la forma de sus cláusulas un programa lógico $\mathbb{P}$ siempre tiene un modelo, por el teorema de Herbrand basta estudiar los modelos de Herbrand de $\mathbb{P}$. El significado declarativo de $\mathbb{P}$ consiste de todas las consecuencias lógicas de dicho programa, las cuales se representan por el modelo mínimo de Herbrand de $\mathbb{P}$.

Proposición 3.5. *Si $\mathbb{P}$ es un programa lógico, entonces $\mathbb{P}$ tiene un modelo de Herbrand.*

Demostración
La base de Herbrand representa al modelo que hace verdaderas a todas las cláusulas atómicas del lenguaje lo cual implica que todas las cláusulas de $\mathbb{P}$ son verdaderas debido a su forma: hechos o reglas.

De modo que cualquier programa lógico tiene al menos un modelo de Herbrand. ■

Proposición 3.6. *Sea $\mathbb{P}$ un programa lógico. $\mathbb{P}$ tiene un modelo mínimo de Herbrand, $\mathcal{M}_\mathbb{P}$ definido por:*

$$\mathcal{M}_\mathbb{P} = \bigcap \{\mathcal{M} \mid \mathcal{M} \text{ es un modelo de Herbrand de } \mathbb{P}\}$$

Demostración
Sabemos que $\mathbb{P}$ tiene al menos un modelo de Herbrand representado por la base de Herbrand. Si $\mathcal{M}_1$ y $\mathcal{M}_2$ son dos modelos de Herbrand de $\mathbb{P}$, su intersección, quien es un conjunto de fórmulas de $\mathcal{B}_\mathcal{L}$, también lo es. De manera que su intersección está bien definida. ■

Proposición 3.7. *Sea $\mathbb{P}$ un programa lógico. El modelo mínimo de Herbrand, $\mathcal{M}_\mathbb{P}$, se representa por el conjunto de átomos que son consecuencia lógica de $\mathbb{P}$, es decir,*

$$\mathcal{M}_\mathbb{P} = \{A \in \mathcal{B}_\mathbb{P} \mid \mathbb{P} \models A\}$$

Demostración

$\subseteq$ Sea $A \in \mathcal{M}_{\mathbb{P}}$. Veamos que $\mathbb{P} \models A$. Supongamos que $\mathbb{P} \not\models A$, entonces $\mathbb{P} \cup \{\neg A\}$ tendría un modelo. Por el teorema de Herbrand, $\mathbb{P} \cup \{\neg A\}$ tendría un modelo de Herbrand $\mathcal{N}$. Entonces $\mathcal{N} \models \neg A$, así $A \notin \mathcal{N}$ pues en $\mathcal{N}$ sólo se encuentran las fórmulas atómicas cerradas que son verdaderas en $\mathcal{N}$. Por otro lado, $\mathcal{N}$ es modelo de Herbrand de $\mathbb{P}$ porque es modelo de $\mathbb{P} \cup \{\neg A\}$. Por minimalidad $\mathcal{M}_{\mathbb{P}} \subseteq \mathcal{N}$, como $A \in \mathcal{M}_{\mathbb{P}}$ también $A \in \mathcal{N}$, lo cual es absurdo ya que habíamos llegado a $A \notin \mathcal{N}$. Suponer $\mathbb{P} \not\models A$ fue una equivocación. Por lo tanto, $\mathbb{P} \models A$.

$\supseteq$ Sea A tal que $\mathbb{P} \models A$. Queremos ver que $A \in \mathcal{M}_{\mathbb{P}}$. Como $\mathcal{M}_{\mathbb{P}}$ es modelo de $\mathbb{P}$, y $\mathbb{P} \models A$, entonces todo modelo que satisface a $\mathbb{P}$ satisface a A, así $\mathcal{M}_{\mathbb{P}}$ satisface a A. Pero A es una fórmula atómica, por consiguiente $A \in \mathcal{M}_{\mathbb{P}}$.

■

Como $\mathcal{M}_{\mathbb{P}}$ contiene exactamente a todos los átomos que son consecuencia lógica de $\mathbb{P}$, este modelo corresponde al significado declarativo de $\mathbb{P}$. Así, la semántica declarativa de $\mathbb{P}$ está dada por $\mathcal{M}_{\mathbb{P}}$.

3.2. Procedimiento para encontrar el modelo mínimo de Herbrand

La semántica declarativa de un programa lógico ha quedado definida mediante el modelo mínimo de Herbrand. A continuación veremos una manera mecánica para construir dicho modelo.

Definición 3.8. Dado un programa lógico $\mathbb{P}$, *el operador de consecuencia inmediata* $\mathcal{T}_{\mathbb{P}}$ se define como:

$$\mathcal{T}_{\mathbb{P}}(\mathcal{K}) =_{\mathsf{def}} \{A \in \mathcal{B}_{\mathbb{P}} \mid A : \text{-}B_1, \ldots, B_n \text{ es instancia cerrada de una cláusula de } \mathbb{P} \text{ y } B_1, \ldots, B_n \in \mathcal{K}\}$$

donde $\mathcal{K} \subseteq \mathcal{B}_{\mathbb{P}}$

Definición 3.9. Sea $\mathbb{P}$ un programa lógico. Definimos las *iteraciones del operador de consecuencia inmediata* $\mathcal{T}_{\mathbb{P}}$ para $\mathcal{K} \subseteq \mathcal{B}_{\mathbb{P}}$ recursivamente como:

$$\mathcal{T}_0(\mathcal{K}) = \mathcal{K}$$
$$\mathcal{T}_{m+1}(\mathcal{K}) = \mathcal{T}_{\mathbb{P}}(\mathcal{T}_m(\mathcal{K}))$$
$$\mathcal{T}_{\omega}(\mathcal{K}) = \bigcup_{i=0}^{\infty} \mathcal{T}_i(\mathcal{K})$$

Proposición 3.10. *Si* $\mathbb{P}$ *es un programa lógico y* $\mathcal{M}_{\mathbb{P}}$ *su modelo mínimo de Herbrand. Entonces,*

$$\mathcal{M}_{\mathbb{P}} = \mathcal{T}_{\omega}(\varnothing)$$

4. Semántica operacional de los programas lógicos

La semántica operacional de un programa lógico $\mathbb{P}$ consiste del conjunto de éxitos, $\mathcal{E}_{\mathbb{P}}$, obtenidos al ejecutar tal programa.

Definición 4.1. El conjunto de éxito de un programa lógico $\mathbb{P}$ se define como

$$\mathcal{E}_{\mathbb{P}} = \{G \in \mathcal{B}_{\mathbb{P}} \text{ cláusula meta} \mid \mathbb{P} \cup \{G\} \text{ tiene una refutación SLD}\}$$

5. Equivalencia de ambas semánticas

Proposición 5.1. *Sea $\mathbb{P}$ un programa lógico. La semánticas declarativa y operacional para $\mathbb{P}$ coinciden. Es decir,*

$$\mathcal{M}_{\mathbb{P}} = \mathcal{E}_{\mathbb{P}}$$

Toda consecuencia lógica *atómica* de $\mathbb{P}$ es un éxito de $\mathbb{P}$, y todo éxito $G \in \mathcal{E}_{\mathbb{P}}$ es consecuencia lógica de $\mathbb{P}$, es decir, $\mathbb{P} \models G$.

6. Incorrectud e incompletud de Prolog

Prolog no cumple con ser correcto ni completo.

6.1. Incorrectud

Prolog reconoce una fórmula como consecuencia lógica de un programa $\mathbb{P}$ cuando ***no*** lo es. Esto es así porque Prolog no verifica las presencias de una variable (X) en un término (t) en la unificación de $\{X, t\}$. Considere[2] el programa con una sola cláusula `data(X,name(X)).` y la siguiente interacción con el intérprete de Prolog.

```
?- trace.
true.

[trace]  ?- data(Y,Y).
   Call: (7) data(_2996, _2996) ? creep
   Exit: (7) data(name(name(name(name(name(name(name(name(name(...)))))))),
          name(name(name(name(name(name(name(name(name(...)))))))))) ? creep
Y = name(Y).
```

Durante la unificación de `data(Y,Y)` con `data(X,name(X))`, comenzamos igualando los primeros argumentos y `X` se sustituye por `Y`. Una vez hecha esta sustitución tramos ahora de unificar `name(Y)` con `Y`. Lo que sucede es un intento de identificar `Y` con `name(Y)` lo que resulta en un nuevo problema, el cual consiste en igualar `name(Y)` con `name(name(Y))` y así sucesivamente. Obtenemos una forma de razonamiento circular que la mayoría de los sistemas de Prolog no pueden resolver.

Para evitar esto es necesario que, cuando se intente la unificación de una variable con un término compuesto, verifiquemos si la variable está contenida dentro de la estructura del término compuesto. Esta verificación se conoce como *verificación de presencias*. Si tratamos de unificar dos términos y llegamos a la unificación de una variable con un término que contiene a dicha variable, entonces la unificación debe fallar.

La mayoría de las implementaciones de Prolog omiten la *verificación de presencias* deliberadamente, principalmente porque es una operación computacionalmente costosa.

Consecuentemente, la meta será exitosa con el unificador `Y=name(Y)` aún cuando debería fallar.

Afortunadamente, SWI-Prolog incorpora un predicado que implementa una unificación correcta, este predicado es `unify_with_occurs_check`. Así la unificación es correcta como se aprecia en los siguientes ejemplos:

[2] http://homepages.inf.ed.ac.uk/pbrna/prologbook/node126.html

```
?- unify_with_occurs_check(X, name(X)).
false.

?- unify_with_occurs_check(X, name(Y)).
X = name(Y).
```

7. Incompletitud

Hay una fórmula que es consecuencia lógica de un programa $\mathbb{P}$ que PROLOG ***no*** reconoce. En el programa de la izquierda, `q` es consecuencia lógica del programa pero PROLOG entra en un ciclo y se queda sin recursos siendo incapaz de poder contestar afirmativamente.

Un reordenamiento de las cláusulas de programa produce una consulta exitosa para `q`, como se muestra del lado derecho.

```
p :- q.
p :- r.
q :- p.
r.

-----------------
?- q.
ERROR: Out of local stack
   Exception: (3,941,684) p ?
```

```
q :- p.
p :- r.
p :- q.
r.

-----------------
?- q.
true
```