

# Representación del Conocimiento

## Lógica de Primer Orden

Dr. Alejandro Guerra-Hernández

**Instituto de Investigaciones en Inteligencia Artificial**  
Universidad Veracruzana

*Campus Sur, Calle Paseo Lote II, Sección Segunda No 112,  
Nuevo Xalapa, Xalapa, Ver., México 91097*

`mailto:aguerra@uv.mx`  
`https://www.uv.mx/personal/aguerra/rc`

Maestría en Inteligencia Artificial 2024



Universidad Veracruzana

- 1 Lógica de Primer Orden
- 2 Programas definitivos
- 3 Resolución-SLD
- 4 Conocimiento en Primer Orden



# Objetivo

- ▶ El uso de la lógica de primer orden para **representar** y **resolver** problemas, nos es familiar por la programación lógica que abordamos en el curso de Programación para la IA.
- ▶ Recuerden que usamos una lógica de primer orden restringida a **cláusulas de Horn** (cláusulas y metas definitivas) y **resolución-SLD**: Prolog.
- ▶ Repasaremos estos conceptos y haremos algunas anotaciones sobre su uso en representación del conocimiento.



# Representación

- ▶ Cuando describimos situaciones de nuestro interés, solemos hacer uso de **enunciados declarativos**.
- ▶ Se trata de expresiones del lenguaje natural que son o bien **verdaderas**, o bien **falsas** (a diferencia de interrogativas, imperativas, etc.).
- ▶ Las proposiciones representan **hechos** que se dan o no en la realidad.
- ▶ La lógica de **primer orden** tienen un compromiso ontológico más fuerte [2], donde la realidad implica además, **objetos** y **relaciones** entre ellos.



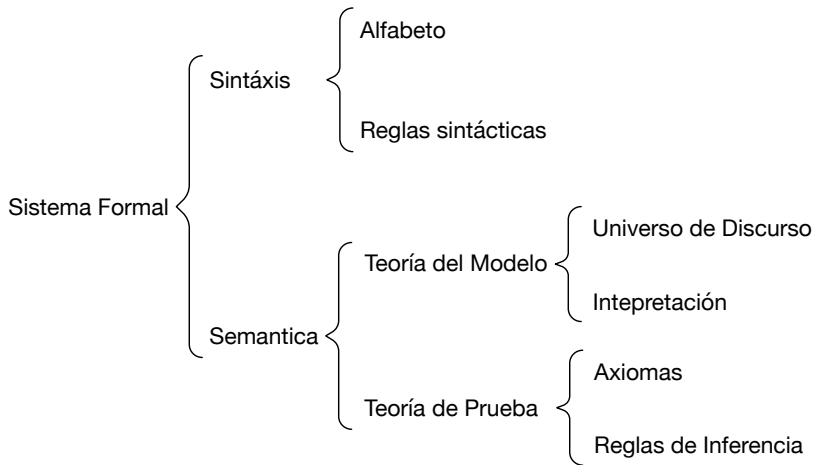
# Razonamiento

- ▶ Consideren los siguientes enunciados declarativos:
  1. Julia es madre y Luis es hijo de Julia.
  2. Toda madre ama a sus hijos.
- ▶ Conociéndolas es posible inferir:
  3. Julia ama a Luis.
- ▶ Para ello definimos un conjunto de **reglas de inferencia**, análogas a las de la deducción natural.
- ▶ Ejemplo:

$$\frac{\phi, \phi \implies \psi}{\psi} (\implies e)$$



# Componentes de un Sistema Formal



# Alfabeto de la Lógica de Primer Orden

*Const* El conjunto de símbolos de constantes;

*Var* El conjunto de símbolos de variables;

*Pred* El conjunto de símbolos de predicados;

*Func* El conjunto de símbolos funcionales ( $Const \subset Func$ );

¬ El operador monario de negación;

∨ El operador binario de disyunción;

∀ El símbolo de cuantificación universal;

() Los paréntesis.



# Reglas sintácticas (términos)

1. Si  $\phi \in Const$  entonces  $\phi \in Term$ ;
2. Si  $\phi \in Var$ , entonces  $\phi \in Term$ ;
3. Si  $\phi/n \in Func$ , entonces  $\phi(\phi_1, \dots, \phi_n) \in Term$  si y sólo si  $\phi_{1 \leq i \leq n} \in Term$ .
4. Ninguna otra expresión es un término.

## ► Ejemplos:

1.  $a, b, c, mesa, \dots$
2.  $X, Y, Z, Cubo1, Cubo2, \dots$
3.  $base(b), sombrero(X), \dots$





# Reglas sintácticas (fórmulas bien formadas)

1. Si  $\phi/n \in Pred$ , entonces  $\phi(\phi_0, \dots, \phi_n) \in \mathcal{L}_{FOL}$  si y sólo si  $\phi_i \in Term, i = 0 \dots n$ ;
2. Si  $\phi \in \mathcal{L}_{FOL}$ , entonces  $\neg\phi \in \mathcal{L}_{FOL}$ ;
3. Si  $\phi \in \mathcal{L}_{FOL}$  y  $\psi \in \mathcal{L}_{FOL}$ , entonces  $(\phi \vee \psi) \in \mathcal{L}_{FOL}$
4. Si  $\phi \in \mathcal{L}_{FOL}$  y  $X \in Vars$  es una variable que ocurre en  $\phi$ , entonces  $\forall X \phi \in \mathcal{L}_{FOL}$
5. Nada más es una fórmula bien formada (fbf).

## ► Ejemplos:

1.  $sobre(a, b), libre(X), ama(X, hijo(X)), \dots$
2.  $\neg libre(a), \neg libre(Y), \dots$
3.  $sobre(X, Y) \wedge \neg libre(X), \dots$
4.  $\forall X ama(X, hijo(X)), \dots$



# Definiciones auxiliares

Conjunción.  $(\phi \wedge \psi) =_{def} \neg(\neg\phi \vee \neg\psi)$ ;

Implicación material.  $(\phi \rightarrow \psi) =_{def} (\neg\phi \vee \psi)$ ;

Equivalencia material.  $(\phi \equiv \psi) =_{def} ((\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi))$ ;

Falso.  $f =_{def} \neg\phi \wedge \phi$ ;

Verdadero.  $t =_{def} \neg f$

Cuantificador existencial.  $\exists X \phi =_{def} \neg(\forall X \neg\phi)$



# Términos en notación BNF

- ▶ Un término se define como:

$$t ::= x \mid c \mid f(t, \dots, t)$$

donde  $x \in Var$ ;  $c \in Func$  tal que  $|c| = 0$ ; y  $f \in Func$  tal que  $|f| > 0$ .



# Fórmulas bien formadas en BNF

- ▶ Las fbf del lenguaje de la Lógica de Primer Orden se construyen como sigue:

$$\phi ::= P(t_1, \dots, t_n) \mid \neg(\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \implies \phi) \mid (\forall x \phi) \mid (\exists x \phi)$$

donde  $P \in Pred$  es un símbolo de predicado de aridad  $n \geq 1$ ;  $t_i$  denota términos; y  $x \in Var$ .



# Notación extra

- ▶ En una fbf de la forma  $\forall X \phi$ , se dice que la fbf  $\phi$  está bajo el **alcance** del cuantificador  $\forall X$ .
- ▶ En tal caso, se dice que la ocurrencia de  $X$  en  $\phi$  está **acotada**, en caso contrario se dice que la ocurrencia de la variable es **libre**.
- ▶ **Ejemplo.** En  $\exists X \text{ sobre}(X, Y)$  la variable  $X$  está acotada, mientras que  $Y$  está libre.
- ▶ Un término sin variables se conoce como **término de base**.
- ▶ **Ejemplo.**  $\text{sobre}(a, b)$ .



# Interpretación

- ▶ Para expresar que al menos hay un bloque que no tiene nada encima, escribimos:  $\exists X \text{ bloque}(X) \wedge \text{libre}(X)$ .
- ▶ Cuando usamos cuantificadores siempre tenemos en mente al  $\mathcal{U}$ , en este caso  $\{a, b, c, d, e, \text{brazo}, \text{mesa}\}$ .
- ▶ Una **interpretación** de esta expresión es un subconjunto de  $\mathcal{U}$  tal que los miembros de ese subconjunto satisfacen el **significado esperado** de la expresión.
- ▶ **Ejemplo.** En este caso  $\{a, e\}$ .



# Teoría del modelo

- ▶ Para obtener un **modelo** para el lenguaje  $\mathcal{L}_{FOL}$  formamos el par  $M = \langle D, V \rangle$ , donde  $D$  es el **universo de discurso** y la **interpretación**  $V$  es una función que satisface las siguientes propiedades:
  - ▶ Si  $\phi \in Const$ , entonces  $V(\phi) = \phi$ ;
  - ▶ Si  $\phi/n \in Pred$ , tal que  $n \geq 1$ , entonces  $V(\phi) \subseteq D^n$ .
- ▶ **Ejemplo.**  $libre^V \subset \{a, b, c, d, e, mesa, mano\}$ .
- ▶ **Ejemplo.**  $sobre^V \subset D \times D$ .
- ▶ Algunas veces la expresión  $V(\phi)$  se abrevia  $\phi^V$ .



# Interpretación para el mundo de bloques

$$a^V = a$$

$$b^V = b$$

$$c^V = c$$

$$d^V = d$$

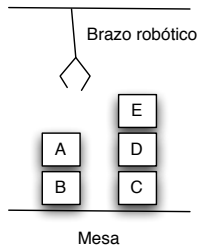
$$e^V = e$$

$$\text{sobre}^V = \{(a, b), (e, d), (d, c)\}$$

$$\text{enLaMesa}^V = \{b, c\}$$

$$\text{libre}^V = \{a, e\}$$

$$\text{porEncima}^V = \{(a, b), (e, d), (e, c), (d, c)\}$$





# Asignación de variables y términos

- ▶ Decimos que  $U$  es una **asignación de variables** basada en el modelo  $M = \langle D, V \rangle$  si para todo  $\phi \in Var$ ,  $U(\phi) \in D$ .
- ▶ La **asignación de términos**  $T$ , dadas la interpretación  $V$  y la asignación de variables  $U$ , es un mapeo de términos a objetos del universo de discurso que se define como sigue:
  1. Si  $\phi \in Const$ , entonces  $T_{VU}(\phi) = V(\phi)$ .
  2. Si  $\phi \in Var$ , entonces  $T_{VU}(\phi) = U(\phi)$ .
  3. Si  $\phi \in Term$  es de la forma  $\psi(\phi_1, \dots, \phi_n)$ ; y  $V(\psi) = g$ ; y  $T_{VU}(\phi_i) = x_i$ , entonces  $T_{VU}(\psi(\phi_1, \dots, \phi_n)) = g(x_1, \dots, x_n)$ .



# Satisfacción

- ▶ Dado un modelo  $M = \langle D, V \rangle$  y una asignación de términos  $T_{VU}$ :
  1.  $\models_V (\phi = \psi)[U]$  ssi  $T_{VU}(\phi) = T_{VU}(\psi)$ .
  2.  $\models_V \phi(\tau_1, \dots, \tau_n)[U]$  ssi  $(T_{VU}(\tau_1), \dots, T_{VU}(\tau_n)) \in \phi^V$ .
  3.  $\models_V (\neg\phi)[U]$  ssi  $\not\models_V \phi[U]$ .
  4.  $\models_V (\phi \wedge \psi)[U]$  ssi  $\models_V \phi[U]$  y  $\models_V \psi[U]$ .
  5.  $\models_V (\phi \vee \psi)[U]$  ssi  $\models_V \phi[U]$  o  $\models_V \psi[U]$ .
  6.  $\models_V (\phi \rightarrow \psi)[U]$  ssi  $\not\models_V \phi[U]$  o  $\models_V \psi[U]$ .
  7.  $\models_V (\forall \nu \phi)[U]$  ssi para todo  $d \in D$  es el caso que  $\models_V \phi[W]$ , donde  $\nu^W = d$  y  $\mu^W = \mu^U$  para  $\mu \neq \nu$ .
  8.  $\models_V (\exists \nu \phi)[U]$  ssi para algún  $d \in D$  es el caso que  $\models_V \phi[W]$ , donde  $\nu^W = d$  y  $\mu^W = \mu^U$  para  $\mu \neq \nu$ .
- ▶ A las asignaciones de variables como  $W$ , se les conoce como  $\nu$ -alternativas.



# Definiciones complementarias

- ▶ Si una interpretación  $V$  satisface a un enunciado  $\phi$  para toda asignación de variables, se dice que  $V$  es un **modelo** de  $\phi$ .
- ▶ Un enunciado se dice **satisfacible** si existe alguna interpretación y asignación de variables que lo satisfaga.
- ▶ Se dice que una fbf  $\phi$  es **válida**, si y sólo si se satisface en toda interpretación y asignación de variables.
- ▶ Las fbf válidas lo son en virtud de su **estructura lógica**, por lo que no proveen información acerca del dominio descrito.
- ▶ **Ejemplo.** Por ejemplo  $p(X) \vee \neg p(X)$  es una fbf válida.



# Mi mamá me ama

► Retomemos el ejemplo de la introducción:

1. Toda madre ama a sus hijos.
2. Julia es madre y Luis es hijo de Julia.
3. Julia ama a Luis.

► Se puede formalizar como:

1.  $\forall X \forall Y \text{ madre}(X) \wedge \text{hijo\_de}(Y, X) \rightarrow \text{ama}(X, Y)$
2.  $\text{madre}(\text{julia}) \wedge \text{hijo\_de}(\text{luis}, \text{julia})$
3.  $\text{ama}(\text{julia}, \text{luis})$



# Reglas de inferencia

- ▶ La inferencia puede verse como un proceso de manipulación de fbf, donde a partir las **premisas**, se producen las **conclusiones**.

- ▶ **Modus Ponens**. O eliminación de la implicación:

$$\frac{\phi \quad \phi \implies \psi}{\psi} \quad (\implies e)$$

- ▶ **Eliminación de cuantificador universal** .:

$$\frac{\forall X \phi(X)}{\phi(t)} \quad (\forall e)$$

- ▶ **Introducción de conjunción** .:

$$\frac{\phi \quad \psi}{\phi \wedge \psi} \quad (\wedge i)$$



# Ejemplo de derivación

▶ Inicio:

1.  $\forall X \forall Y \text{madre}(X) \wedge \text{hijo\_de}(Y, X) \implies \text{ama}(X, Y)$
2.  $\text{madre}(\text{julia}) \wedge \text{hijo\_de}(\text{luis}, \text{julia})$

▶ Al aplicar la eliminación de cuantificador universal ( $\forall E$ ) a (1) obtenemos:

3.  $\forall Y (\text{madre}(\text{julia}) \wedge \text{hijo\_de}(Y, \text{julia}) \implies \text{ama}(\text{julia}, Y))$

▶ Al aplicar nuevamente ( $\forall E$ ) a (3) obtenemos:

4.  $\text{madre}(\text{julia}) \wedge \text{hijo\_de}(\text{luis}, \text{julia}) \implies \text{ama}(\text{julia}, \text{luis})$

▶ Finalmente, al aplicar Modus Ponens a (2) y (4):

5.  $\text{ama}(\text{julia}, \text{luis})$



# Solidez y robustez

- ▶ Un conjunto de reglas de inferencia se dice **sólido**, si para todo conjunto de fbf cerradas (sin ocurrencia de variables libres)  $\Delta$  y cada fbf cerrada  $\phi$ , siempre que  $\Delta \vdash \phi$  se tiene que  $\Delta \models \phi$ .
- ▶ Las reglas de inferencia se dicen **completas** si cuando  $\Delta \models \phi$  siempre es el caso que  $\Delta \vdash \phi$ .



# Enunciados declarativos

- ▶ Describen relaciones **positivas** entre elementos de  $\mathcal{U}$ : Incondicionadas (**hechos**) y Condicionadas (**reglas**).

1. Antonio es hijo de Juan.
2. Ana es hija de Antonio.
3. Juan es hijo de Marcos.
4. Alicia es hija de Juan.
5. El nieto de una persona es el hijo del hijo de esa persona.

1.  $hijo\_de(antonio, juan)$
2.  $hijo\_de(ana, antonio)$
3.  $hijo\_de(juan, marcos)$
4.  $hijo\_de(alicia, juan)$
5.  $\forall X \forall Y (nieto\_de(X, Y) \leftarrow \exists Z (hijo\_de(\bar{Z}, Y) \wedge hijo\_de(X, Z)))$





# Formas alternativas para una regla

- ▶ La fórmula:

$$\forall X \forall Y (\text{nieto\_de}(X, Y) \leftarrow \exists Z (\text{hijo\_de}(Z, Y) \wedge \text{hijo\_de}(X, Z)))$$

- ▶ Se puede escribir como:

- ▶  $\forall X \forall Y (\text{nieto\_de}(X, Y) \vee \neg \exists Z (\text{hijo\_de}(Z, Y) \wedge \text{hijo\_de}(X, Z)))$

- ▶  $\forall X \forall Y (\text{nieto\_de}(X, Y) \vee \forall Z \neg (\text{hijo\_de}(Z, Y) \wedge \text{hijo\_de}(X, Z)))$

- ▶  $\forall X \forall Y \forall Z (\text{nieto\_de}(X, Y) \vee \neg (\text{hijo\_de}(Z, Y) \wedge \text{hijo\_de}(X, Z)))$

- ▶  $\forall X \forall Y \forall Z (\text{nieto\_de}(X, Y) \leftarrow (\text{hijo\_de}(Z, Y) \wedge \text{hijo\_de}(X, Z)))$

- ▶ Con **equivalencias** como:

- ▶  $\phi \rightarrow \psi \equiv \neg \phi \vee \psi$  ó

- ▶  $\forall X \phi \equiv \neg \exists X \neg \phi.$



# Literales

- ▶ Una **literal** es un átomo o la negación de un átomo.
- ▶ Una **literal positiva** es un átomo.
- ▶ Una **literal negativa** es la negación de un átomo.
- ▶ Ejemplos:
  - ▶  $hijo\_de(juan, marcos)$ .
  - ▶  $\neg hijo\_de(juan, alicia)$ .
- ▶ Son los bloques de construcción  $\phi_i$  en:

$$\phi_0 \leftarrow \phi_1 \wedge \cdots \wedge \phi_n \quad (n \geq 0)$$



# Cláusulas

- ▶ Una **cláusula** es una disyunción finita de cero o más literales.
- ▶ Una cláusula se dice **definitiva**, si tiene exactamente una literal positiva.

$$\phi_0 \vee \neg\phi_1 \vee \cdots \vee \neg\phi_n \quad (n \geq 0)$$

- ▶ Lo cual es equivalente a la forma general de fbf que nos interesaba:

$$\phi_0 \leftarrow \phi_1 \wedge \cdots \wedge \phi_n \quad (n \geq 0)$$

- ▶ La cláusula vacía (sin literales) se denota como  $\square$  y es equivalente a  $\square \leftarrow \blacksquare$  (en nuestra notación previa  $\perp \leftarrow \top$ ).



# Hechos y reglas revisitados

- ▶ En  $\phi_0 \leftarrow \phi_1 \wedge \cdots \wedge \phi_n$  ( $n \geq 0$ ), tenemos que:
  - ▶ Si  $n = 0$ , la literal  $\phi_0$  será positiva, por lo que la cláusula definitiva será un **hecho**.
  - ▶ Si  $n > 0$  la cláusula definitiva toma la forma de una **regla**, donde  $\phi_0$  es la **cabeza** de la regla; y la conjunción  $\phi_1 \wedge \cdots \wedge \phi_n$  su **cuerpo**.
- ▶ Una forma restringida de **cuantificación**:

$$\forall X \forall Y (\text{nieto\_de}(X, Y) \vee \neg \exists Z (\text{hijo\_de}(Z, Y) \wedge \text{hijo\_de}(X, Z))).$$



# Programa y Meta Definitivos

- ▶ Un **programa definitivo** es un conjunto finito de cláusulas definitivas.
- ▶ Una cláusula sin literales positivas es una **meta definitiva**.

$$\leftarrow \phi_1 \wedge \cdots \wedge \phi_n \quad (n \geq 1)$$



# Ejemplos

- Considere las siguientes consultas:

Consulta	Meta definitiva
¿Es Ana hija de Antonio?	← <i>hijo(ana, antonio)</i>
¿Quién es nieto de Ana?	← <i>nieto(X, ana)</i>
¿De quién es nieto Antonio?	← <i>nieto(antonio, X)</i>
¿Quién es nieto de quién?	← <i>nieto(X, Y)</i>



# Cláusulas de Horn

- ▶ Una **cláusula de Horn** es una cláusula ó una meta definitivas.
- ▶ La cláusula vacía  $\square$  es una meta definitiva y, por lo tanto, una cláusula de Horn.
- ▶ Las cláusulas de Horn implican restricciones expresivas.
- ▶ **Ejemplo** No podemos expresar  $p(a) \vee p(b)$
- ▶ Debido a su estructura restringida, las cláusulas de Horn son **más fáciles** de manipular que las cláusulas generales.



# Significado lógico de las metas

- ▶ El significado lógico de las metas puede explicarse haciendo referencia a la fbf equivalente cuantificada universalmente:

$$\forall X_1 \dots X_m \neg(\phi_1 \wedge \dots \wedge \phi_n)$$

donde todas las  $X_i$  ocurren en la meta.

- ▶ Equivale a:

$$\neg \exists X_1 \dots X_m (\phi_1 \wedge \dots \wedge \phi_n)$$





# Conocimiento positivo

- ▶ Los programas definitivos solo pueden expresar conocimiento positivo, tanto los hechos, como las reglas, nos dicen que elementos de una estructura están en una relación, pero **no nos dicen cuales no**.
- ▶ Por lo tanto, al usar el lenguaje de los programas definitivos, no es posible construir **descripciones contradictorias**, i.e., conjuntos de fbf no satisfacibles.
- ▶ **Todo programa definitivo tiene un modelo.**



# Modelos e interpretaciones (recordatorio)

- ▶ Sea  $\phi$  una fbf y  $V$  una interpretación.  $V$  es un **modelo** de  $\phi$  si  $\models_V \phi$ .
- ▶ Sea  $\Delta$  un conjunto finito de fbf y  $V$  una interpretación.  $V$  es un modelo de  $\Delta$  si  $\models_V \phi$  para toda  $\phi \in \Delta$ .
- ▶ Una clase interesante de modelos para los programas definitivos se conoce como **interpretaciones de Herbrand**.



# Universo y Base de Herbrand

- ▶ Sea  $L$  un alfabeto extraído de un programa definitivo  $\Delta$ , donde  $|Const| \geq 1$ .
- ▶ El **Universo de Herbrand** ( $U_\Delta$ ) es el conjunto de todos los términos formados con las constantes y funtores de  $L$ .
- ▶ La **Base de Herbrand** ( $B_\Delta$ ) es el conjunto de todos los átomos que pueden formarse con los predicados y los términos en  $U_L$ .



# Ejemplo: impar

- ▶ Sea  $\Delta = \{ \text{impar}(s(0)), \text{impar}(s(s(X))) \leftarrow \text{impar}(X) \}$ .
- ▶  $U_{\Delta} = \{0, s(0), s(s(0)), s(s(s(0))), \dots\}$
- ▶  $B_{\Delta} = \{ \text{impar}(0), \text{impar}(s(0)), \text{impar}(s(s(0))), \dots \}$



# Interpretación de Herbrand

- ▶ Sea  $\Delta$  un programa definitivo y  $L$  el alfabeto compuesto por los símbolos de  $\Delta$ .
- ▶  $V$  es una interpretación de Herbrand de  $\Delta$ , si y sólo si:
  - ▶ El dominio de  $V$  es  $U_\Delta$ .
  - ▶ Para cada constante  $c \in L$ ,  $c^V = c$ .
  - ▶ Para cada functor  $f/n \in L$ ,  $f^V : U_\Delta^n \mapsto U_\Delta$ .
  - ▶ Para cada predicado  $p/n \in L$ ,  $p^V \subseteq U_\Delta^n$ .



# Modelo de Herbrand

- ▶ Sea  $\Delta$  un programa definitivo;
- ▶  $L$  el alfabeto compuesto por los símbolos en  $\Delta$ ;
- ▶ y  $V$  una interpretación de Herbrand de  $\Delta$ .
- ▶ Si  $V$  es un modelo de toda fbf en  $\Delta$ , se dice que es un **modelo de Herbrand** de  $\Delta$ .



# Ejemplo: impar

- ▶ Consideren el programa  $\Delta$  en el ejemplo *impar*/1.
- ▶ Una posible interpretación de este programa es

$$\textit{impar}^V = \{s(0), s(s(s(0)))\}$$

- ▶ Una interpretación de Herbrand se puede especificar mediante una familia de tales relaciones (una por cada símbolo de predicado).



# Resultados de interés

- ▶ Para poder determinar si una interpretación de Herbrand  $V$  es un **modelo de una fbf** cuantificada universalmente  $\forall\phi$ , es suficiente verificar si  $\phi$  es verdadera en  $V$ , para todas las asignaciones posibles de las variables de  $\phi$ .
- ▶ Para el lenguaje restringido de cláusulas definitivas, si queremos verificar que una fbf atómica  $\phi$  es **consecuencia de un programa definitivo**  $\Delta$  basta con verificar que todo modelo de Herbrand de  $\Delta$  es también un modelo de Herbrand de  $\phi$ .





# Programa definitivo extendido con meta definitiva

- ▶ Sea  $\Delta$  un programa definitivo y  $\gamma$  una meta definitiva.
- ▶ Sea  $L$  un alfabeto compuesto por los símbolos en el programa y la meta definitivos.
- ▶ Si  $V'$  es un modelo de  $\Delta \cup \{\gamma\}$ , entonces  $V = \{\phi \in B_{\Delta} \mid \models_{V'} \phi\}$  es un modelo de Herbrand de  $\Delta \cup \{\gamma\}$ .



# Consistencia

- ▶ Sea  $\Delta$  un programa definitivo y  $\phi$  una cláusula definitiva.
- ▶ Sea  $\Delta' = \Delta \cup \{\neg\phi\}$ .
- ▶ Entonces  $\Delta \models \phi$  si y sólo si  $\Delta'$  no tiene modelo de Herbrand.
- ▶ Esto es, si  $\Delta'$  es **no satisfacible**, lo cual es cierto sólo si  $\Delta'$  no tiene modelos y por lo tanto, no tiene modelo de Herbrand.



# Intersección de modelos de Herbrand

- ▶ Sea  $M$  una familia **no vacía** de modelos de Herbrand de un programa definitivo  $\Delta$ .
- ▶ Entonces la intersección  $V = \bigcap M$  es un modelo de Herbrand de  $\Delta$ .
- ▶ Al tomar la intersección de los modelos de Herbrand de un programa definitivo (todos tienen al menos un modelo, e.g.,  $B_\Delta$ ), obtenemos el **modelo mínimo de Herbrand**.



# Ejemplo

- ▶ Sea  $\Delta$  el programa definitivo  $\{masculino(adan), femenino(eva)\}$  con su interpretación obvia.
- ▶  $\Delta$  tiene los siguientes modelos de Herbrand:
  1.  $\{masculino(adan), femenino(eva)\}$
  2.  $\{masculino(adan), masculino(eva), femenino(eva)\}$
  3.  $\{masculino(adan), masculino(eva), femenino(adan)\}$
  4.  $\{masculino(adan), masculino(eva), femenino(eva), femenino(adan)\}$
- ▶ La intersección de los modelos nos lleva a un modelo de Herbrand.
- ▶ El modelo mínimo es el único que corresponde con el **modelo pretendido** del programa.



# Consecuencia Lógica

- ▶ El modelo mínimo de Herbrand de un programa definitivo  $\Delta$ ,  $M_\Delta$ , es el conjunto de todas las **consecuencias lógicas atómicas de base** del programa:

$$M_\Delta = \{\phi \in B_\Delta \mid \Delta \models \phi\}$$

- ▶ La prueba de este teorema pasa por demostrar que  $M_\Delta \supseteq \{\phi \in B_\Delta \mid \Delta \models \phi\}$  y que  $M_\Delta \subseteq \{\phi \in B_\Delta \mid \Delta \models \phi\}$ .



# Programas y metas definitivos

- ▶ Consideren el siguiente programa definitivo  $\Delta$ :

*papa(juan, marta).*

*recien\_nacido(marta).*

*orgullosa(X) ← padre(X, Y), recién\_nacido(Y).*

*padre(X, Y) ← papa(X, Y).*

*padre(X, Y) ← mama(X, Y).*

- ▶ ¿Cómo interpretamos la meta  $\leftarrow orgullosa(Z).$  ?



# Metas

- ▶  $\leftarrow orgullososo(Z) \equiv \forall Z \neg orgullososo(Z)$
- ▶  $\forall Z \neg orgullososo(Z) \equiv \neg \exists Z orgullososo(Z)$ .
- ▶ Si demostramos que ese enunciado es contradictorio en  $\Delta$ , entonces sabremos que  $\Delta \models \exists Z orgullososo(Z)$ .
- ▶ Pero eso solo respondería *true* a la pregunta original.
- ▶ El objetivo en realidad es encontrar una **substitución**  $\theta$  tal que el conjunto  $\Delta \cup \{\neg orgullososo(Z)\theta\}$  sea insatisfacible, lo que equivale a que  $\Delta \models orgullososo(Z)\theta$ .
- ▶ **Ejemplo.**  $\Delta \cup \neg orgullososo(Z)\{Z/juan\}$



# Razonamiento

- ▶ Asumimos la meta  $\gamma_0$  – Para todo  $Z$ ,  $Z$  no está orgulloso.
- ▶ Una regla en  $\Delta$  describe una condición para que alguien esté orgulloso:

$$\text{orgullosa}(X) \leftarrow \text{padre}(X, Y), \text{recien\_nacido}(Y)$$

- ▶ Lo cual es lógicamente equivalente a:

$$\forall (\neg \text{orgullosa}(X) \rightarrow \neg(\text{padre}(X, Y) \wedge \text{recien\_nacido}(Y)))$$





# Estrategia

- ▶ Partiendo de:

$$\forall (\neg \text{orgullosa}(X) \rightarrow \neg(\text{padre}(X, Y) \wedge \text{recien\_nacido}(Y)))$$

- ▶ Al renombrar  $X$  por  $Z$ , eliminar el cuantificador universal y usar *modus ponens* con respecto a  $\gamma_0$ , obtenemos  $\gamma_1$ :

$$\neg(\text{padre}(Z, Y) \wedge \text{recien\_nacido}(Y))$$

- ▶ o su equivalente:

$$\leftarrow \text{padre}(Z, Y), \text{recien\_nacido}(Y).$$

- ▶  $\gamma_1$  que es verdadera en todo modelo  $\Delta \cup \{\gamma_0\}$ .



# Resolución

- Ahora solo queda probar que  $\Delta \cup \{\gamma_1\}$  es no satisfacible. Observen que  $\gamma_1$  es equivalente a la fbf:

$$\forall Z \forall Y (\neg \text{padre}(Z, Y) \vee \neg \text{recien\_nacido}(Y))$$

- $\gamma_1$  no es satisfacible para  $\Delta$ , si en todo modelo de  $\Delta$  hay una persona que es padre de un recién nacido:

$$\text{padre}(X, Y) \leftarrow \text{papa}(X, Y).$$

- Por lo que  $\gamma_1$  se reduce a  $\gamma_2$ :

$$\leftarrow \text{papa}(Z, Y), \text{recien\_nacido}(Y).$$



# Estrategia recursiva

- ▶ El programa declara que *juan* es padre de *marta*:

$papa(juan, marta).$

- ▶ Así que sólo resta probar que “*marta* no es una recién nacida” no se puede satisfacer junto con  $\Delta$ :

$\leftarrow recien\_nacido(marta).$

- ▶ pero el programa contiene el hecho:

$recien\_nacido(marta).$

- ▶ equivalente a  $\neg recien\_nacido(marta) \rightarrow \square$
- ▶ lo que conduce a una refutación  $\square$ .



# Resumiendo

- ▶ Para probar la existencia de algo:
  - ▶ Suponer lo **opuesto**
  - ▶ y usar **modus ponens** y la regla de **eliminación del cuantificador universal**,
  - ▶ para encontrar un **contra ejemplo** al supuesto.



# Unificador

- ▶ Una meta es un conjunto de átomos a ser probados.
- ▶ Seleccionamos un átomo de la meta  $p(s_1, \dots, s_n)$  y una cláusula del programa con la forma  $p(t_1, \dots, t_n) \leftarrow A_1, \dots, A_n$  si encontramos una **substitución**  $\theta$  que hace que  $p(s_1, \dots, s_n)\theta$  y  $p(t_1, \dots, t_n)\theta$  sean idénticos.
- ▶ Tal sustitución se conoce como **unificador**.
- ▶ La nueva meta se construye reemplazando el átomo seleccionado en la meta original, por los átomos de la cláusula seleccionada, aplicando  $\theta$  a todos los átomos obtenidos de esta manera.



# Propiedades

- ▶ Si se prueba en  $k$  pasos que la meta definitiva en cuestión no puede satisfacerse, probamos que:

$$\leftarrow (\phi_1, \dots, \phi_m)\theta_1 \dots \theta_k$$

- ▶ es una instancia que no puede satisfacerse. Por tanto:

$$\Delta \models (\phi_1 \wedge \dots \wedge \phi_m)\theta_1 \dots \theta_k$$



# Observaciones I

- ▶ Esta computación **no es determinista**: Cualquier átomo de la meta puede ser seleccionado y pueden haber varias cláusulas del programa que unifiquen con el átomo seleccionado.
- ▶ Pueden existir **unificadores alternativos** para dos átomos.
- ▶ Es posible que el átomo seleccionado **no unifique** con ninguna cláusula en el programa, i.e., que no sea posible construir un contra ejemplo para la meta.
- ▶ La computación puede caer en un **ciclo**, sin producir solución alguna.



# Observaciones II

- ▶ Computar  $subtermino(X, t)$  (**verificación de ocurrencia**) hace que el algoritmo sea altamente ineficiente.
- ▶ El standard ISO Prolog (1995) declara que la unificación es **no decidable**.
- ▶ Al eliminar la verificación de ocurrencia es posible que al intentar resolver  $X = f(X)$  obtengamos  $X = f(f(X)) \cdots = f(f(f \dots))$ .





# Formalizando

- El método de razonamiento descrito informalmente al inicio de esta sesión, puede resumirse con la siguiente regla de inferencia:

$$\frac{\forall \neg(\phi_1 \wedge \dots \wedge \phi_{i-1} \wedge \phi_i \wedge \phi_{i+1} \wedge \dots \wedge \phi_m) \quad \forall(\psi_0 \leftarrow \psi_1 \wedge \dots \wedge \psi_n)}{\forall \neg(\phi_1 \wedge \dots \wedge \phi_{i-1} \wedge \psi_1 \wedge \dots \wedge \psi_n \wedge \phi_{i+1} \wedge \dots \wedge \phi_m)\theta}$$

- donde:

1.  $\phi_1, \dots, \phi_m$  son fbf atómicas.
2.  $\psi_0 \leftarrow \psi_1, \dots, \psi_n$  es una cláusula definitiva en el programa  $\Delta$  ( $n \geq 0$ ).
3.  $MGU(\phi_i, \psi_0) = \theta$ .



# Observaciones

- ▶ La regla tiene dos premisas: una **meta** y una **cláusula definitiva**.
- ▶ El alcance de los cuantificadores es **disjunto**.
- ▶ Solo hay un cuantificador universal para la conclusión. Se requiere que el conjunto de variables en las premisas sea **disjunto**.
- ▶ Puesto que todas las variables en las premisas están cuantificadas, es siempre posible **renombrar** las variables de la cláusula definitiva para cumplir con esta condición.



# S de selección

- ▶ La meta definida puede incluir muchas fbf atómicas que unifican con la cabeza de alguna cláusula en el programa.
- ▶ Es deseable contar con un mecanismo determinista para seleccionar un átomo  $\phi_i$  a unificar.
- ▶ Se asume una función que selecciona una submeta de la meta definida (**función de selección**).



# Usando la resolución-SLD

- ▶ El punto de partida es una meta definida  $\gamma_0$ :

$$\leftarrow \phi_1, \dots, \phi_m \quad (m \geq 0)$$

- ▶ Una submeta  $\phi_i$  será seleccionada. Una nueva meta  $\gamma_1$  se construye al seleccionar una cláusula del programa  $\psi_0 \leftarrow \psi_1, \dots, \psi_n$  ( $n \geq 0$ ) cuya cabeza  $\psi_0$  unifica con  $\phi_i$ , resultando en  $\theta_1$ .  $\gamma_1$  tiene la forma:

$$\leftarrow (\phi_1, \dots, \phi_{i-1}, \psi_1, \dots, \psi_n, \dots, \phi_m)\theta_1$$

- ▶ Ahora es posible aplicar el principio de resolución a  $\gamma_1$  para obtener  $\gamma_2$ , y así sucesivamente.



# Terminación

- ▶ El proceso puede terminar o no. Hay dos situaciones donde no es posible obtener  $\gamma_{i+1}$  a partir de  $\gamma_i$ :
  1. cuando la submeta seleccionada no puede ser resuelta (no es unificable con la cabeza de una cláusula del programa).
  2. cuando  $\gamma_i = \square$  (meta vacía = f).



# Derivación-SLD

- ▶ Sea  $\gamma_0$  una meta definitiva,  $\Delta$  un programa definitivo y  $\mathcal{R}$  una función de selección. Una **derivación SLD** de  $\gamma_0$  (usando  $\Delta$  y  $\mathcal{R}$ ) es una secuencia finita o infinita de metas:

$$\gamma_0 \xrightarrow{\phi_0} \gamma_1 \dots \gamma_{n-1} \xrightarrow{\phi_{n-1}} \gamma_n$$

- ▶  $\phi_i \in \Delta$ . Las variables en  $\phi_i$  se renombran con subíndices  $i$ .



# Substitución computada

- ▶ Cada derivación SLD nos lleva a una secuencias de MGUs  $\theta_1, \dots, \theta_n$ .  
La composición

$$\theta = \begin{cases} \theta_1\theta_2 \dots \theta_n & \text{si } n > 0 \\ \epsilon & \text{si } n = 0 \end{cases}$$

de MGUs se conoce como la **substitución computada** de la derivación.



# Ejemplo

- ▶ Consideren la meta definida  $\leftarrow orgullososo(Z)$  y el programa del inicio de esta clase. Entonces

$$\gamma_0 = \leftarrow orgullososo(Z).$$

$$\phi_0 = orgullososo(X_0) \leftarrow padre(X_0, Y_0), recién\_nacido(Y_0).$$

- ▶ La unificación de  $orgullososo(Z)$  y  $orgullososo(X_0)$  nos da el MGU  $\theta_1 = \{X_0/Z\}$ . Asumamos que nuestra función de selección es tomar la **submeta más a la izquierda**:

$$\gamma_1 = \leftarrow padre(Z, Y_0), recién\_nacido(Y_0).$$

$$\phi_1 = padre(X_1, Y_1) \leftarrow papa(X_1, Y_1).$$

con  $\theta_2 = \{X_1/Z, Y_1/Y_0\}$ .





# Ejemplo

- ▶ La derivación continua como sigue:

$$\gamma_2 = \leftarrow \text{papa}(Z, Y_0), \text{recien\_nacido}(Y_0).$$

$$\phi_2 = \text{papa}(\text{juan}, \text{marta}).$$

$$\gamma_3 = \leftarrow \text{recien\_nacido}(\text{marta}).$$

$$\phi_3 = \text{recien\_nacido}(\text{marta}).$$

$$\gamma_4 = \square$$

- ▶ la sustitución computada para esta derivación es:

$$\begin{aligned} \theta_1\theta_2\theta_3\theta_4 &= \{X_0/Z\}\{X_1/Z, Y_1/Y_0\}\{Z/\text{juan}, Y_0/\text{marta}\}\epsilon \\ &= \{X_0/\text{juan}, X_1/\text{juan}, Y_1/\text{marta}, \\ &\quad Z/\text{juan}, Y_0/\text{marta}\} \end{aligned}$$



# Refutación-SLD y derivación fallida

- ▶ Una derivación SLD finita:

$$\gamma_0 \xrightarrow{\phi_0} \gamma_1 \cdots \gamma_{n-1} \xrightarrow{\phi_{n-1}} \gamma_n$$

donde  $\gamma_n = \square$ , se llama **refutación SLD** de  $\gamma_0$ .

- ▶ Una derivación de la meta  $\gamma_0$  cuyo último elemento no es la meta vacía y no puede resolverse con ninguna cláusula del programa, es llamada **derivación fallida**.

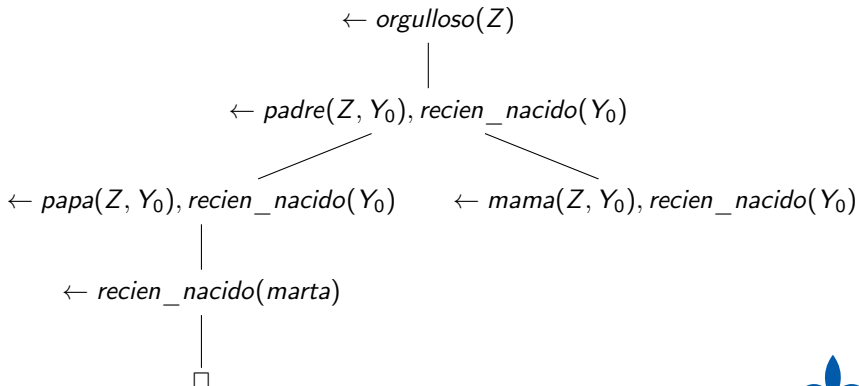


# Arbol-SLD

- ▶ Sea  $\Delta$  un prog. definitivo,  $\gamma_0$  una meta definitiva, y  $\mathcal{R}$  una función de selección. El **árbol-SLD** de  $\gamma_0$  (usando  $\Delta$  y  $\mathcal{R}$ ) es un árbol etiquetado, posiblemente infinito, que cumple con:
  - ▶ La raíz del árbol está etiquetada por  $\gamma_0$ .
  - ▶ Si el árbol contiene un nodo etiquetado como  $\gamma_i$  y existe una cláusula renombrada  $\phi_i \in \Delta$  tal que  $\gamma_{i+1}$  es derivada de  $\gamma_i$  y  $\phi_i$  via  $\mathcal{R}$ , entonces el nodo etiquetado como  $\gamma_i$  tiene un hijo etiquetado  $\gamma_{i+1}$ . El arco entre ambos es  $\phi_i$ .



## Ejemplo



# Propiedades de la resolución-SLD

**Solidez.** Sea  $\Delta$  un programa definitivo,  $\mathcal{R}$  una función de selección, y  $\theta$  una sustitución de respuesta computada a partir de  $\Delta$  y  $\mathcal{R}$  para una meta  $\leftarrow \phi_1, \dots, \phi_m$ . Entonces  $\forall ((\phi_1 \wedge \dots \wedge \phi_m)\theta)$  es una consecuencia lógica del programa  $\Delta$ .

**Completez.** Sea  $\Delta$  un programa definitivo,  $\mathcal{R}$  una función de selección y  $\leftarrow \phi_1, \dots, \phi_m$  una meta definitiva. Si  $\Delta \models \forall ((\phi_1 \wedge \dots \wedge \phi_m)\sigma)$ , entonces existe una refutación de  $\leftarrow \phi_1, \dots, \phi_m$  vía  $\mathcal{R}$  con una sustitución de respuesta computada  $\theta$ , tal que  $(\phi_1 \wedge \dots \wedge \phi_m)\sigma$  es un caso de  $(\phi_1 \wedge \dots \wedge \phi_m)\theta$ .



# Base de conocimientos

- ▶ Ahora podemos explorar en detalle el proceso de creación de una **base de conocimientos** ( $\Delta$ ).
- ▶ Recuerden que el conocimiento implica asumir que el mundo satisface alguna propiedad, expresada como un **enunciado declarativo**.
- ▶  $\Delta$  está formada por una colección de tales enunciados y nosotros asumimos que las proposiciones expresadas por ellos son las **creencias** de un agente putativo.



# Consideraciones

- ▶ ¿Qué es lo que queremos (o nuestro agente quiere) **computar**?
- ▶ Establecer las **razones** por las que la inferencia es necesaria en nuestros sistemas y el **número de veces** que debe llevarse a cabo.
- ▶ Establecer una **ontología**:
  - ▶ Las clases de **objeto** que nos interesan;
  - ▶ Las **propiedades** que esos objetos pueden tener;
  - ▶ Las **relaciones** que puedan darse entre ellos.
- ▶ A este proceso que se orienta a  $\Delta$  desde el nivel del conocimiento, se le conoce como **Ingeniería del Conocimiento** [1].



# Vocabulario

- ▶ Comenzar por el conjunto de **predicados** y **funciones** dependientes del dominio.
- ▶ ¿Qué clases de **objetos** habrá en nuestro sistema?
- ▶ Las constantes serán usadas para representar **individuos con nombre**:  
Ej. **alejandro\_guerra**, **rc**, etc.
- ▶ Es posible que necesitemos **múltiples identificadores**:  
Ej. 6183 puede denotar al mismo individuo que **alejandro\_guerra**.
- ▶ **Otros** individuos con nombre: Entidades legales, lugares, objetos.  
Ej. **iphone15**, **iiii**, etc.





# Tipos de objetos

- ▶ Ahora será necesario establecer los **tipos** de objetos que emergen de los individuos con nombre adoptados.
- ▶ Para ello solemos usar **predicados** de aridad uno:  
Ej. `no_personal(6183)`, `prof(nicandro_cruz)`, `inst(iiia)`, etc.



# Atributos

- ▶ Los predicados unarios también pueden representar **propiedades** de nuestros objetos:  
Ej. `sni(6183)`, `sni(nicandro_cruz)`, `pnpc(mia)`, etc.
- ▶ Observen que **no podemos distinguir** entre atributos y tipos de objeto. Si esto es necesario, el lenguaje FOL podría extenderse.



# Relaciones y funciones

- ▶ Las **relaciones** están representadas como predicados  $n$ -arios:  
Ej. `adscripcion(6183, iiii)`, `prof(rc, 6183)`, etc.
- ▶ No olviden que hay relaciones que **no son binarias**:  
Ej. `horario(rc, 10, 12, [martes, jueves])`, etc.
- ▶ Las **funciones** pueden tomar varios argumentos, pero suelen ser unarias:  
Ej.  $s(6) \mapsto 7$ , etc.
- ▶ Todas las funciones se asumen **totales**, si hay alguien sin sucesor en el dominio, deberíamos considerar definir  $s/1$  como una relación binaria:  
Ej.  $s(6, 7)$ , etc.



# Hechos y reglas

- ▶ Con este vocabulario nuestros hechos simples pueden representarse como **literales**, i.e., predicados atómicos o su negación:  
Ej. `prof(6183)`, `¬jubilado(6183)`, etc.
- ▶ Estos hechos constituyen una **base de datos** que podría almacenarse como una **tabla relacional**.
- ▶ Otros hechos básicos son los que tiene que ver con **igualdad**:  
Ej. `martin_aguilar == Rector(uv)`, etc.
- ▶ También podemos definir reglas:  
Ej. `investigador_en(Invest,Inst) :- sni1(Invest), inst(Inst).`



# Reglas terminológicas I

**Disjunto.** Dos predicados son disjuntos si la neg. de uno implica al otro:

Ej. `menor_edad(X) :- \+ mayor_edad(X)`.

**Subtipo.** Un predicado subsume al otro:

Ej. `medico(X) :- cirujano(X)`.

**Exhaustivo.** Dos o más tipos completan el concepto:

Ej. `sni(X) :- candidato(X); sni1(X); sni2(X); sni3(X);  
emerito(X)`.

**Simétrico.** Definen una relación simétrica:

Ej. `colega(X,Y) :- colega(Y,X)`.

**Inverso.** Definen una relación inversa:

Ej. `padre(X,Y) :- hijo(Y,X)`.



# Reglas terminológicas II

**Restricción.** Establecen una restricción de tipo de objeto:

Ej. `profEn(Prof, iiia) :- sni(Prof).`

**Definición.** Definen un término:

Ej. `matutino(Curso) :- horario(Curso, _, Fin, _), Fin < 14.`



# Individuos abstractos

- ▶ Consideren las posibles representaciones de que `nic` compró una `bici`:
  - ▶ `compra(nic, bici)`
  - ▶ `compra(nic, bici, costco)`
  - ▶ `compra(nic, bici, costco, 12000)`
  - ▶ `compra(nic, bici, costco, 12000, feb14)`
  - ▶ etc.
- ▶ Solución: Definir un individuo abstracto y tantas relaciones binarias o funciones unarias como sean necesarias:  
Ej. `compra(f9872). precio(f9872,12000).`, etc.
- ▶ Permite: `precioEnDolares(C, PUs) :- precio(C, PMx), PUs = PMx/19.80`



# Otros hechos

**Estadísticos.** Incluyen información sobre la probabilidad o la proporción de individuos que satisfacen el predicado.

Ej. La mayoría de los empleados está de vacaciones.

**Defaults.** Incluye características que normalmente son razonables de asumir, al menos que uno sea advertido de lo contrario.

Ej. Las aves vuelan.

**Intencionales.** Actitudes proposicionales.

Ej. Este SNI cree que es hora de irse de vacaciones y lo desea.





# Referencias I

- [1] RJ Brachman y HJ Levesque. *Knowledge representation and reasoning*. San Francisco, CA, USA: Morgan Kaufmann Publishers, 2004.
- [2] SJ Russell y P Norvig. *Artificial Intelligence: A Modern Approach*. Third. Prentice Hall Series in Artificial Intelligence. USA: Prentice Hall, 2009.

