DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Lógica en Inteligencia Artificial

© Fernando Berzal, berzal@acm.org

Lógica en Inteligencia Artificial

- **Modelos lógicos**
  - Lógica Proposicional
  - Lógica de Predicados
- **Razonamiento lógico**
  - Mecanismos de inferencia
    - Modus ponens
    - Modus tollens
    - Resolución
  - Demostración por refutación
  - Motores de inferencia
- **Otros modelos lógicos**

Lógica

Modelos lógicos de representación del conocimiento

Representación formal de las relaciones existentes entre objetos (y entre los objetos y sus propiedades):

Los modelos lógicos clásicos más conocidos son:

- La Lógica Proposicional
- La Lógica de Predicados

Básicamente, se diferencian en que la primera no admite argumentos en los predicados mientras que la segunda sí.

Lógica Proposicional

- Se utilizan proposiciones que representan afirmaciones, que pueden ser verdaderas o falsas.
- Las proposiciones se unen con operadores lógicos ($\wedge$ [y], $\vee$ [o], $\neg$ [no]), y se construyen reglas con el operador de implicación lógica ($\rightarrow$).
- Existen mecanismos de inferencia que permiten obtener nuevos datos a partir de los datos ya conocidos (p.ej., modus ponens, modus tollens…)

Lógica Proposicional

EJEMPLO: Modus ponens

p	si p
$p \rightarrow q$	y p implica q
q	entonces q

es un razonamiento válido porque q siempre será verdad, independientemente de lo que represente, cuando se cumplan p y $p \rightarrow q$.

Lógica Proposicional

EJEMPLO: Modus ponens

p = "hace calor"
q = "el profesor está incómodo"

Memoria de trabajo (datos):	p
Base de conocimiento (reglas):	$p \rightarrow q$
Deducción (aplicando modus ponens):	q

- q pasa a formar parte de la memoria de trabajo.

Lógica Proposicional

EJEMPLO: Modus tollens

$\neg q$	si no q
$p \rightarrow q$	y p implica q
$\neg p$	entonces no p

es un razonamiento válido porque p siempre será falso, independientemente de lo que represente, cuando se no se cumpla q y se verifique $p \rightarrow q$.

Lógica Proposicional

EJEMPLO: Modus tollens

p = "hace calor"
q = "el profesor está incómodo"

Memoria de trabajo (datos) : $\neg q$
Base de conocimiento (reglas): $p \rightarrow q$
Deducción (aplicando modus tollens): $\neg p$

¿seguro?

Lógica Proposicional

LIMITACIONES

El emparejamiento requiere igualdad exacta del antecedente de la regla con los hechos conocidos.

p = "hombre"
q = "mortal"
x = "todo hombre es mortal"
y = "Juan es hombre"

¡¡¡ No podemos deducir nada !!!

Lógica Proposicional

SOLUCIÓN: **Lógica de Predicados**

Necesitamos establecer una relación entre objetos (personas) y propiedades (esHombre, esMortal):

esHombre(Juan)
$\forall x$ (esHombre(x) $\rightarrow$ esMortal(x))

Podemos deducir: esMortal(Juan)

Lógica de Predicados

La Lógica de Predicados añade
la posibilidad de utilizar **cuantificadores**:

- ∀ (para todo)
- ∃ (existe)

Mecanismos de inferencia:
modus ponens, modus tollens, resolución…

EJEMPLO: Lenguaje de programación PROLOG

Lógica de Predicados

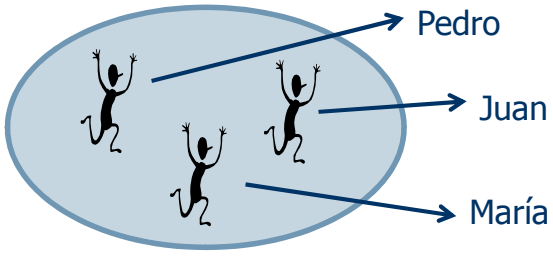
Formalización de un modelo en Lógica de Predicados:

Sintaxis: Constantes

- Mediante símbolos nombramos objetos

  p.ej. Juan, Pedro, María…

- El conjunto de todos esos símbolos se denomina dominio de discurso.



Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:

Sintaxis: Predicados

- Mediante símbolos representamos relaciones entre objetos (y entre objetos y sus propiedades):

  p.ej. esHombre/1 /1 ≡ 1 argumento
  quiere/2 /2 ≡ 2 argumentos

- Los predicados reciben términos como argumentos:

  p.ej. esHombre(Juan)
  quiere(Juan, María)

Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:

Semántica: Concepto de interpretación

En una interpretación,
se realiza una correspondencia entre los objetos y relaciones del mundo real y los objetos definidos en la sintaxis de la Lógica de Predicados.

Hipótesis de Mundo Cerrado
[CWA: Closed World Assumption]

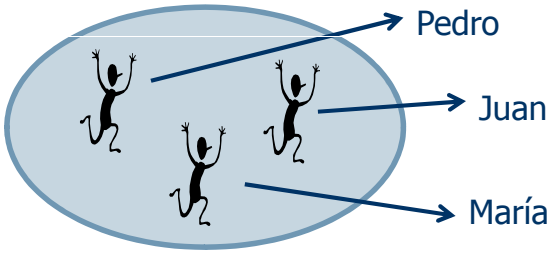
Todo aquello que no se incluya
se considerará que es falso.

Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:
Semántica: Concepto de interpretación

EJEMPLO:

Dominio



Relaciones

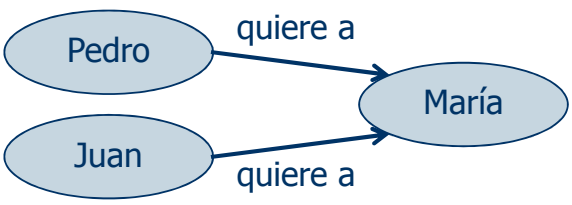
a. Ser hombre esHombre/1
b. Querer a otra persona quiere/2

Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:
Semántica: Concepto de interpretación

Mediante predicados representamos hechos:

esHombre(Juan)
esHombre(Pedro)
quiere(Pedro,María)
quiere(Juan,María)



Diremos, p.ej., que esHombre(Juan) es verdadero, mientras que esHombre(María) es falso.

Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:

Sintaxis: Funciones

"Devuelven" un valor del dominio y podemos interpretarlas como una forma avanzada de nombrar objetos.

p.ej. padre/1

quiere(Juan, "el padre de Juan")

⇓

quiere(Juan, padre(Juan))

Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:

Sintaxis: Variables

Las variables se usarán como comodines que pueden ser sustituidos por objetos del dominio.

p.ej. x, y…

Por ahora, sólo consideraremos que son símbolos que forman parte de la sintaxis de la Lógica de Predicados.

Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:

Sintaxis: Términos

- Una constante es un término.
- Una variable es un término.
- Si f es una función n-aria y t1 ... tn son términos, entonces f(t1 , ... , tn) es un término.

En general,

los términos son los argumentos de los predicados.

p.ej. Juan, padre(Juan), padre(x)...

Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:

Sintaxis: Átomos y conectivos lógicos

- Si P es un símbolo de predicado y t1 ... tn son términos, entonces P(t1, ... , tn) es un **átomo**.

  Los átomos corresponden a los datos de nuestra base de conocimiento (la forma de representar hechos en Lógica de Predicados).

- Los conectivos lógicos ($\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$) nos permitirán conectar átomos para representar las reglas de nuestra base de conocimiento (en forma de f.b.f.).

Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:

Sintaxis: Cuantificación universal y existencial

$\forall x$ esHombre(x)

- Será verdad cuando todos los objetos del dominio de discurso satisfagan el predicado esHombre.
- Si en nuestro dominio de discurso hay objetos "no hombres" (p.ej. Gatos), será falso.

$\exists x$ esHombre(x)

- Será verdad cuando exista algún objeto del dominio de discurso que satisfaga el predicado esHombre.
- Con tener un hombre en el dominio de discurso, será cierto, aunque también haya gatos…

Lógica de Predicados

Formalización de un modelo en Lógica de Predicados:

Sintaxis: Fórmulas bien formadas (f.b.f.)

- Un átomo es una f.b.f. (atómica).
- Si F y G son f.b.f.,
  entonces $\neg F$, $F \wedge G$, $F \vee G$, $F \rightarrow G$, $F \leftrightarrow G$ son f.b.f.
- Si F es una f.b.f. y x una variable,
  entonces (F), $(\forall x)F$ y $(\exists x)F$ son f.b.f.

Lógica de Predicados

Ámbito de los cuantificadores

El ámbito de un cuantificador es la f.b.f. sobre la que se aplica, p.ej.

$$(\forall x)\ (\ (\forall y)P(x,y) \vee (\exists z)R(x,z)\ )$$

ámbito de y: $(\forall y)P(x,y)$ — ámbito de z: $(\exists z)R(x,z)$

ámbito de x: $(\forall x)\ (\ (\forall y)P(x,y) \vee (\exists z)R(x,z)\ )$

Lógica de Predicados

Equivalencias lógicas (≡)

En las siguientes expresiones, se supone que P, Q contienen la variable x en alguno de sus argumentos:

$\forall x\ \neg P \equiv \neg\exists x\ P$ $\qquad$ $\neg\forall x\ P \equiv \exists x\ \neg P$

$\forall x\ P \equiv \neg\exists x\ \neg P$ $\qquad$ $\exists x\ P \equiv \neg\forall x\ \neg P$

$\neg P \wedge \neg Q \equiv \neg(P \vee Q)$ $\qquad$ $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$

$P \wedge Q \equiv \neg(\neg P \vee \neg Q)$ $\qquad$ $P \vee Q \equiv \neg(\neg P \wedge \neg Q)$

Lógica de Predicados

Literales y cláusulas

- Un literal es un átomo o la negación de un átomo.
- Una cláusula es una disyunción de literales.

SÍ esHombre(x)
quiere(Juan,Maria) $\vee$ quiere(Juan,Elena)
$\neg$quiere(Juan,Pedro)

NO esHombre(x) $\wedge$ quiere(x,y)
quiere(Juan,Elena) $\wedge$ $\neg$quiere(Juan,Pedro)

Lógica de Predicados

Cláusulas de Horn

Una cláusula de Horn sólo admite un literal **NO** negado.

P

$\neg$P

$\neg$ P $\vee$ $\neg$ Q $\vee$ $\neg$ R $\vee$ S $\equiv$ **P $\wedge$ Q $\wedge$ R $\rightarrow$ S**

- Las cláusulas de Horn nos permiten representar hechos o reglas con un único consecuente.
- PROLOG sólo admite cláusulas de Horn.

Lógica de Predicados

Asignación de un grado de verdad a una f.b.f.

A partir del grado de verdad de los átomos (P, Q), podemos determinar el grado de verdad de cualquier f.b.f. utilizando la siguiente tabla:

P	Q	P∨Q	P∧Q	¬P	P→Q	P↔Q
T	T	T	T	F	T	T
T	F	T	F	F	F	F
F	T	T	F	T	T	F
F	F	F	F	T	T	T

NOTA. P ∨ Q será verdad en una interpretación I, si alguno de ellos es verdad en I.

Lógica de Predicados

Asignación de un grado de verdad a una f.b.f.

Formalmente, esto nos lleva a casos como:

Si P→Q es verdadero y resulta que P es falso, entonces ¡Q es verdadero!

En I.A., se parte siempre de reglas y premisas (antecedentes) verdaderas, a partir de los cuales se construye el grado de verdad del consecuente.

Ejemplos

"Todos las madres quieren a sus hijos"

Solución incorrecta:

$\forall x \, \forall y \;\; madre(x,y) \wedge quiere(x,y)$

Respuesta correcta:

$\forall x \, \forall y \;\; madre(x,y) \rightarrow quiere(x,y)$

Si sabemos que María es la madre de Juan, esto es madre(María, Juan), podemos aplicar la sustitución {x/María, y/Juan} para concluir quiere(María, Juan).

Ejemplos

"Todos los padres quieren a sus hijos"

- ¿Qué pasa si en el dominio de discurso tenemos información del tipo padre(GatoFelix,GatoIsidoro)?
- Concluiremos quiere(GatoFelix,GatoIsidoro), lo cual no tiene por qué ser verdad.

Solución:

Usar un predicado esPersona:

$\forall x \, \forall y \; (padre(x,y) \wedge esPersona(x) \rightarrow quiere(x,y))$

Ejemplos

¿Qué representa la siguiente regla?

$\forall x \, \forall y \, (esPersona(x) \rightarrow quiere(x,y))$

- Supongamos como hecho esPersona(Juan):
  Entonces, podemos concluir $\forall y \, (quiere(Juan,y))$
  ¡Juan quiere a todos (y a todo objeto del dominio)!

- Esto es, todas las personas lo quieren todo ¿?

Ejemplos

¿Qué representarían las siguiente reglas?

- $\forall x \, \exists y \, (esPersona(x) \rightarrow quiere(x,y))$

- $\forall x \, \exists y \, (esPersona(x) \rightarrow quiere(x,y) \wedge esPersona(y))$

- $\forall x \, \forall y \, (madre(x,y) \rightarrow quiere(y,x))$

- $\forall x \, \forall y \, (esPersona(x) \wedge tiene(x,y) \rightarrow esFeliz(x))$

Ejemplos

"Una persona gobierna a todos los guatemaltecos"

Solución incorrecta:

$\exists x \, \forall y \, (esGuatemalteco(y) \wedge gobierna(x,y))$

Respuestas correctas:

$\exists x \, \forall y \, (esGuatemalteco(y) \rightarrow gobierna(x,y))$

$\forall y \, (esGuatemalteco(y) \rightarrow gobierna(PresidenteGuatemala,y))$

Ejemplos

Es fundamental pensar en el tipo de objetos que se van a representar, ya que esto determinará el tipo de predicados que será necesario definir, p.ej.

"Todos los guatemaltecos tienen un perro"

$\forall x \, (esGuatemalteco(x) \rightarrow tienePerro(x))$

Tenemos varias opciones:

- tienePerro/1
- tieneAnimal/2 + esPerro/1
- tiene/2 + esPerro/1

Ejemplos

Para decidir entre una representación u otra, debemos preguntarnos:

- ¿Existirán reglas sobre los perros en general? "Todos los perros ladran" (mejor usar esPerro/1).
- ¿Existirán reglas sobre la propiedad "tener" en general? "Todos los que tienen algo son felices" (mejor usar tener/2)
- ¿Existirán reglas que atañen sólo a la propiedad en el sentido de "tenerAnimal"? (mejor usar tenerAnimal/2)

Ejemplos

"Todos los que tienen un animal deben vacunarlo"

- Usando tieneAnimal/2:

  $\forall x \, \forall y \, (\text{tieneAnimal}(x,y) \rightarrow \text{debeVacunar}(x,y))$

- Usando tiene/2:

  $\forall x \, \forall y \, (\text{tiene}(x,y) \wedge \text{esPerro}(y) \rightarrow \text{debeVacunar}(x,y))$
  $\forall x \, \forall y \, (\text{tiene}(x,y) \wedge \text{esGato}(y) \rightarrow \text{debeVacunar}(x,y))$
  …

Ejemplos

Alternativa A

Los animales no son objetos del dominio

tieneAnimal(Pedro)

tieneAnimal(Heidi)

$\forall x$ (tieneAnimal(x) $\rightarrow$ debeDesparasitarASuAnimal(x))

Los animales no son objeto del dominio,
por lo que no podríamos especificar la regla
"todos los guatemaltecos tienen un perro"
indicando cuál es el perro que tiene cada uno.

Ejemplos

Alternativa B

Aun no siendo objetos del dominio, distinguimos tipos

tieneGato(Pedro)

tienePerro(Heidi)

$\forall x$ (tienePerro(x) $\rightarrow$ debeVacunarRabia(x))

$\forall x$ (esGuatemalteco(x) $\rightarrow$ tienePerro(x))

Ejemplos

Para desparasitar al conjunto de los animales:

- Incluimos una regla para cada tipo de animal:
  $\forall x$ (tienePerro(x) $\rightarrow$ debeDesparasitarASuAnimal(x))
  $\forall x$ (tieneGato(x) $\rightarrow$ debeDesparasitarASuAnimal(x))

- O, mucho mejor, usamos una única regla:
  $\forall x$ (tieneAnimal(x) $\rightarrow$ debeDesparasitarASuAnimal(x))
  E introducimos reglas que relacionen tienePerro/1 y tieneGato/1 con tieneAnimal/1:
  $\forall x$ (tienePerro (x) $\rightarrow$ tieneAnimal(x))
  $\forall x$ (tieneGato (x) $\rightarrow$ tieneAnimal (x))

Ejemplos

Alternativa C

Los animales son objetos del dominio, sin distinguir tipos

tieneAnimal(Pedro,ElGatoDePedro)
tieneAnimal(Heidi,Niebla)

$\forall x \forall y$ (tieneAnimal(x,y) $\rightarrow$ debeVacunar(x,y))

Ejemplos

Alternativa D

Los animales son objetos del dominio, distinguiendo tipos

tieneAnimal(Heidi,Niebla)
esPerro(Niebla)
esGato(Misifú)

$\forall x \; \forall y$ (tieneAnimal(x,y) $\rightarrow$ debeVacunar(x,y))
$\forall x$ (esPerro(x) $\rightarrow$ ladra(x))

Ejemplos

"Cada dueño debe vacunar de la rabia a su(s) perro(s)"

- Usando debeVacunarRabia/2:

$\forall x \; \forall y$ tieneAnimal(x,y) $\wedge$ esPerro(y)
$\rightarrow$ debeVacunarRabia(x,y)

- Usando debeVacunarRabiaASusPerros/1:

$\forall x \; \forall y$ tieneAnimal(x,y) $\wedge$ esPerro(y)
$\rightarrow$ debeVacunarRabiaASusPerros(x)

Razonamiento lógico

En vez de partir de una interpretación
(en la que sólo existen hechos: P, Q…)
y derivar consecuencias lógicas de ellos (P→Q),
en Inteligencia Artificial:

Se parte de un conjunto de hechos y reglas (P, P→Q)
y se obtienen nuevas conclusiones (Q)
que sean consistentes con lo que se tenía
(**razonamiento deductivo**).

Mecanismos de inferencia

Modus ponens

P	si P
P → Q	y P implica Q
Q	entonces Q

Es decir, si B.C. = {P, P→Q}, aplicando M.P. podemos asegurar que Q es verdad, independientemente de la interpretación, por lo que obtendríamos una nueva base de conocimiento:

$$B.C. = \{P, P \rightarrow Q, Q\} \equiv \{P \wedge (P \rightarrow Q) \wedge Q\}$$

Mecanismos de inferencia

Modus ponens en Lógica de Predicados

En Lógica de Predicados
necesitamos igualar el antecedente con el hecho:

$$\frac{P(a) \qquad \forall x\,(P(x) \rightarrow Q(x))}{Q(a)}$$

Esto se consigue **unificando** los átomos P(x) y P(a)
mediante la **sustitución** $u = \{ x/a \}$

Mecanismos de inferencia

Unificación

La aplicación de una sustitución u a una f.b.f. F,
lo notaremos por Fu.

$$P(x)u = P(a)$$
$$Q(x)u = Q(a)$$
$$(P(x) \rightarrow Q(x))u = (P(a) \rightarrow Q(a))$$

De esta forma, los átomos P(x) y P(a) se unifican aplicándoles la sustitución $u = \{ x/a \}$:

$$P(x)u = P(a)u = P(a)$$

La sustitución u se denomina **unificador**.

Mecanismos de inferencia

Unificación

- Para igualar dos f.b.f. podrían usarse varias posibles sustituciones.

  p.ej. P(x) y P(y) podrían unificarse con la sustitución

  u = {x/a , y/a}

  o también con la sustitución

  u = {x/y}

- Lo lógico sería aplicar la segunda sustitución, que es la menos restrictiva (unificador de máxima generalidad).

Mecanismos de inferencia

Modus ponens en Lógica de Predicados

- ¿Es siempre correcta la regla de inferencia del modus ponens en Lógica de Predicados, independientemente de la interpretación considerada? SÍ

- Por eso decimos que el modus ponens es una regla de inferencia válida.

  Una regla de inferencia no válida sería: Si P(a) y $(\exists x)P(x)$, entonces $(\forall x)P(x)$, aunque en alguna interpretación sí podría ser correcta. Por ejemplo, si el dominio de P es {a}.

Mecanismos de inferencia

Modus tollens

$\neg Q$	si no Q
$P \rightarrow Q$	y P implica Q
$\neg P$	entonces no P

es una regla de inferencia válida porque P siempre será falso, independientemente de lo que represente, cuando se no se cumpla Q y se verifique $P \rightarrow Q$.

Mecanismos de inferencia

Resolución

$$\frac{P \vee Q \qquad \neg P \vee R}{Q \vee R}$$

También es una regla de inferencia válida, que además abarca MP y MT como casos particulares.

Mecanismos de inferencia

Forma normal clausal

- Para poder realizar inferencias usando resolución es necesario que las f.b.f. vengan en un formato adecuado: la forma normal clausal (prenexa).

- Básicamente, una base de conocimiento está expresada en forma normal clausal cuando se expresa como una conjunción de cláusulas, sin cuantificadores existenciales y con todos los cuantificadores universales están a la izquierda de cada cláusula de forma que las negaciones afecten únicamente a los términos.

Mecanismos de inferencia

Forma normal clausal

- Existe un algoritmo para transformar cualquier conjunto de f.b.f. en forma normal clausal.

EJEMPLO $\quad quiere(Juan,Maria) \vee (quiere(Juan,Elena) \wedge \neg quiere(Juan,Pedro))$

se transformaría en:

$(quiere(Juan,Maria) \vee quiere(Juan,Elena))$

$\wedge$

$(quiere(Juan,Maria) \vee \neg quiere(Juan,Pedro))$

Mecanismos de inferencia

Forma normal clausal

Base de conocimiento (f.b.f.)
$BC = \{FBF_1, FBF_2 \ldots FBF_n\}$

↓ Transformación f.b.f.→f.n.c.

Base de conocimiento (f.n.c.)
$BC = \{FN_1, FN_2 .. FN_n\}$

↓ Resolución

Base de conocimiento (f.n.c.)
$BC = \{FN_1, FN_2 \ldots FN_n, FN_{n+1} \ldots\}$

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.

Constantes de Skolem

$\exists x\, P(x)$ nos indica que existe un elemento del dominio de discurso para el que se cumple el predicado P.

Dicho elemento no tiene por qué ser conocido, pero al menos existe uno. Lo llamamos SK, por ejemplo, donde SK es una constante de Skolem.

p.ej. sustituiremos $\exists x\, P(x)$ por $P(SK)$
y $\exists x\, Q(x)$ por $Q(SK_2)$

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.
Constantes de Skolem

EJEMPLO: Existen personas que quieren a sus padres

$\exists x$ quiere(x,padre(x))
$\exists x \exists y$ (padre(x,y) $\wedge$ quiere(y,x))

Podemos dar un nombre a esa(s) persona(s) que sabemos que existe(n) pero no sabemos su nombre:

quiere($SK_{alguien}$, padre($SK_{alguien}$))
padre(SK_{padre}, SK_{hijo}) $\wedge$ quiere(SK_{hijo},SK_{padre})

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.
Constantes de Skolem

EJEMPLO: "Todos los que quieren son felices"

Podemos aplicar la regla a las constantes de Skolem del ejemplo anterior:

- Regla: $\forall p \forall c$ (quiere(p,c) → esFeliz(p))
- Hecho: quiere(SK_{hijo}, SK_{padre})
- Conclusión: esFeliz(SK_{hijo})

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.
Constantes de Skolem

EJEMPLO ABSURDO: $\exists x\ \exists y\ (padre(x,y) \rightarrow quiere(y,x))$

Existen al menos dos valores para los que se cumple la regla, llamémoslos SK_{padre} y SK_{hijo}:
$padre(SK_{padre}, SK_{hijo}) \rightarrow quiere(SK_{hijo}, SK_{padre})$

Existen dos personas desconocidas para las que, si la primera es padre de la segunda, entonces la segunda la querría. No tiene mucho sentido.

Conclusión: No será usual encontrar reglas afectadas únicamente con cuantificación existencial.

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.
Funciones de Skolem

¿Se sustituye $\forall x\ \exists y\ P(x,y)$ por $\forall x\ P(x,SK)$?
¡NO! Cada x tiene asociado un y distinto.

EJEMPLO: "Cualquier persona tiene un padre"
$\forall x\ \exists y\ padre(y,x)$
$\forall x\ padre(desconocido, x)$ MAL

Lo correcto es usar $\forall x\ P(SK(x),x)$, donde SK es una función de Skolem: $\forall x\ padre(elPadreDe(x),x)$.

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.

Funciones de Skolem

Si una variable cuantificada existencialmente, y, está dentro del ámbito de varias variables cuantificadas universalmente x_1 ... x_n, se sustituirá y por una función $f(x_1,...,x_n)$

NOTA: Si en nuestro dominio representamos otros tipos de objetos, usaríamos una regla del tipo:

$\forall x$ esPersona(x) → padre(elPadreDe(x),x)

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.

Constantes y funciones de Skolem

EJEMPLOS

- Todos los padres quieren a todos sus hijos:

  $\forall x \forall y$ (padre(x,y) → quiere(x,y))

- Existen padres que quieren a todos sus hijos:

  $\exists x \forall y$ (padre(x,y) → quiere(x,y))

  $\forall y$ (padre(SK,y) → quiere(x,y))

- Cada padre quiere a alguno de sus hijos:

  $\forall x \exists y$ (padre(x,y) → quiere(x,y))

  $\forall x$ (padre(x, elHijoAmadoDe(x))
  → quiere(x, elHijoAmadoDe(x)))

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.

1. Eliminar el conectivo → usando la equivalencia
   $A \rightarrow B \equiv \neg A \vee B$

2. Aplicar la negación sólo sobre los átomos. Para ello, usamos las reglas siguientes:
   $\neg\neg F \equiv F$
   $\neg(F \vee G) \equiv \neg F \wedge \neg G$
   $\neg(F \wedge G) \equiv \neg F \vee \neg G$
   $\neg((\forall x)\ F[x]) \equiv (\exists x)\ (\neg F[x])$
   $\neg((\exists x)\ F[x]) \equiv (\forall x)\ (\neg F[x])$

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.

3. Normalizar variables (cambiar el nombre de aquellas variables que estén en ámbitos distintos):

   $\forall x\ P(x) \vee \forall x\ Q(x) \equiv \forall x\ P(x) \vee \forall z\ Q(z)$
   $\forall x\ (P(x) \vee Q(x))$ permanece igual
   $\forall x\ P(x) \wedge \forall x\ Q(x) \equiv \forall x\ P(x) \wedge \forall z\ Q(z)$
   $\forall x\ (P(x) \wedge Q(x))$ permanece igual

   $\exists x\ P(x) \vee \exists x\ Q(x) \equiv \exists x\ P(x) \vee \exists z\ Q(z)$
   $\exists x\ (P(x) \vee Q(x))$ permanece igual
   $\exists x\ P(x) \wedge \exists x\ Q(x) \equiv \exists x\ P(x) \wedge \exists z\ Q(z)$
   $\exists x\ (P(x) \wedge Q(x))$ permanece igual

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.

4. Eliminar la cuantificación existencial, usando constantes y funciones de Skolem, p.ej.

   $\forall x\ \exists y\ P(x,y) \equiv \forall x\ P(x,f_{SK6}(x))$
   $\exists z\ Q(z) \equiv Q(c_{SK7})$

5. Mover los cuantificadores (universales) a la izquierda:

   $\forall x\ (P(x) \vee (\forall z)\ Q(x,z)) \equiv \forall x\ \forall z\ (P(x) \vee Q(x,z))$

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.

6. Suprimir el prefijo de la cuantificación, p.ej.
   $\forall x\ \forall z\ (P(x) \vee Q(x,z))$ se transforma en $P(x) \vee Q(x,z)$
   y se entiende que la cuantificación es universal.

7. Convertir la f.b.f. en conjunción de disyunciones, utilizando la propiedad distributiva

   $(F \wedge G) \vee H \equiv (F \vee H) \wedge (G \vee H)$

   y eliminando paréntesis de los términos disyuntivos

   $(F \vee G) \vee H \equiv F \vee G \vee H$

Mecanismos de inferencia

Algoritmo de transformación f.b.f. → f.n.c.

8. Eliminar los símbolos $\wedge$ para formar un conjunto de cláusulas: $(F \wedge G \wedge H) \equiv \{F , G , H\}$.

9. Volver a normalizar (cambiar de nombre) las variables, de forma que no aparezca la misma variable en dos cláusulas distintas.
   $\forall x (P(x) \wedge Q(x)) \equiv \forall x P(x) \wedge \forall x Q(x)$
   $\equiv \forall x P(x) \wedge \forall z Q(z)$, por lo que podemos justificar el cambio de nombre de las variables de las cláusulas $\{P(x), Q(x)\}$ a $\{P(x),Q(z)\}$.

Demostración por refutación

Hasta ahora hemos deducido.

Ahora, queremos demostrar.

Base de Conocimiento:

B.C. = $\{F1, ..., Fn\} \equiv \{F1 \wedge .... \wedge Fn\}$

Objetivo:

$\{G\}$

Método (demostración por refutación):

Demostrar que B.C. $\cup \{\neg G\} = \{F1 \wedge ... \wedge Fn \wedge \neg G\}$ es inconsistente; es decir, que no hay ninguna interpretación en la que todas las f.b.f. se cumplan.

Demostración por refutación

Si podemos demostrar que {F1 ∧ ... ∧ Fn ∧ ¬G} es falso y partimos de la hipótesis de que la B.C. es verdadera, la única conclusión posible es que ¬G sea falso o, lo que es lo mismo, que G es verdadero.

NOTA: B.C. ∪ {¬G} representa unión de conjuntos; es decir, {A} ∪ {B} es {A, B}, que corresponde realmente a la cláusula A∧B.

Demostración por refutación

Refutación por resolución

B.C. = {P , ¬P∨Q} ≡ {P ∧ (¬P∨Q)}
Objetivo Q

Demostramos que {P∧(¬P∨Q)∧ ¬Q} es inconsistente:

- A partir de P y ¬P∨Q obtenemos Q por resolución.
- A partir de Q y ¬Q obtenemos □ por resolución, donde □ representa la cláusula vacía, que indica que existe una contradicción.

Hemos demostrado que B.C. ∪ {¬Q} es inconsistente, luego podemos asegurar que Q es una consecuencia lógica de la base de conocimiento B.C.

Demostración por refutación

Refutación por resolución

La idea es que, con resolución,
podemos demostrarlo todo.

Pero, ¿qué es demostrarlo todo?

Supongamos que B.C.={P(A)}. Por sí sola, la resolución no puede generar la cláusula $P(A) \vee \neg P(A)$ y, sin embargo, ésta es cierta en cualquier interpretación (es una tautología). Por tanto, la resolución no va a generar todo aquello que sea demostrable.

Demostración por refutación

Refutación por resolución

- Pero si decimos explícitamente que queremos demostrar una f.b.f. concreta, por ejemplo $P(A) \vee \neg P(A)$, entonces podemos utilizar refutación por resolución y demostrar que B.C. $\cup \{\neg(P(A) \vee \neg P(A)\}$ es inconsistente.
- La resolución es completa para refutación; es decir, si especificamos una f.b.f. concreta, F, y ésta es una consecuencia lógica de un conjunto de f.b.f., entonces la resolución nos dirá que B.C.$\cup\{\neg F\}$ es inconsistente: **¡no necesitamos ningún otro mecanismo de inferencia!**

Demostración por refutación

Refutación por resolución

Nos queda un tema por tratar:

¿cómo de rápido puede conseguirse lo anterior mediante un programa de ordenador?

Pueden construirse motores de inferencia que usen resolución utilizando distintas estrategias, pero todos ellos tienen forzosamente un tiempo de ejecución exponencial en el peor caso.

Demostración por refutación

Refutación por resolución

- Supongamos que queremos demostrar que F **SÍ** es una consecuencia lógica de la base de conocimiento: Sabemos que la resolución será teóricamente capaz de demostrarlo (aunque no sepamos cuánto tardará).

- Supongamos que queremos demostrar que F **NO** es una consecuencia lógica de la base de conocimiento:
  Existen algunas fórmulas F para las cuales no hay ningún mecanismo de inferencia (incluida la resolución) que concluya que F no es demostrable.

Demostración por refutación

Refutación por resolución

La Lógica de Predicados es semidecidible:

El programa estaría ejecutándose indefinidamente sin parar y no sabemos si es porque aún no hemos esperado lo suficiente para encontrar la demostración (siendo ésta posible) o porque nunca nos dará un resultado (esto es, porque no es demostrable).

Sólo sabemos que, cuando pare,
es porque ha encontrado una demostración.

Demostración por refutación

Refutación por resolución

CONSIDERACIONES FINALES

- Modus Ponens es completo por refutación para bases que sólo tengan cláusulas de Horn.
- La Lógica de Predicados sí es decidible para cláusulas de Horn (las usadas en PROLOG). Esto quiere decir que, aplicando un mecanismo de inferencia completo, siempre podremos decir si una f.b.f. es una consecuencia lógica o no de la base de conocimiento. En cualquier caso, el tiempo requerido puede ser excesivo [Ginsberg, capítulo 8].

Motores de inferencia

Supongamos la siguiente base de conocimiento:

$$BC = \{A \vee B\ ,\ \neg A \vee C\ ,\ P\ ,\ P \rightarrow Q\}$$

Podríamos aplicar distintos mecanismos de inferencia deductivos, como por ejemplo:

$\{A \vee B\ ,\ \neg A \vee C\ ,\ P\ ,\ P \rightarrow Q\}$

$\Downarrow$ **resolución**

$\{A \vee B\ ,\ \neg A \vee C\ ,\ P\ ,\ P \rightarrow Q,\ \mathbf{B \vee C}\}$

$\Downarrow$ **modus ponens**

$\{A \vee B\ ,\ \neg A \vee C\ ,\ P\ ,\ P \rightarrow Q\ ,\ B \vee C,\ \mathbf{Q}\}$

Motores de inferencia

- En el ejemplo empleamos razonamiento monótono (problema ignorable), por lo que bastaría usar una estrategia de control irrevocable.
- El motor de inferencia podría, por ejemplo, intentar aplicar primero Modus Ponens antes de Modus Tollens (en el caso de que no quiera usarse la resolución).
- También podemos optar por usar sólo la resolución como único mecanismo de inferencia.

Motores de inferencia

- Si usamos resolución, podemos elegir entre varias estrategias, p.ej.

  Estrategia de conjunto soporte

  Intuitivamente, trataría de resolver primero con el objetivo que pretendemos demostrar o con cláusulas que provienen de dicho objetivo.

  Otras estrategias utilizan funciones de evaluación heurísticas [Ginsberg, capítulo 9].

Otros modelos lógicos

Existen otras lógicas que permiten incorporar aspectos como el tiempo, la incertidumbre, que hechos dejen de ser ciertos…

- Lógicas modales
- Lógicas temporales
- Lógica difusa [fuzzy logic]
- Lógicas no monótonas
- …

Otros modelos lógicos

Lógicas descriptivas [DL: Description Logic]

Más expresivas que la lógica proposicional, pero limitándose a fragmentos decidibles de la lógica de predicados de primer orden.

La base formal de los lenguajes de descripción de ontologías que se usan en la Web Semántica, p.ej. **OWL** [Web Ontology Language].

Otros modelos lógicos

Lógicas descriptivas [DL: Description Logic]

- **ABox** [assertion component]: Hechos.

```
A es una instancia de B.
Juan es una persona.
```

- **TBox** [terminological component]: Descripción de un sistema mediante un vocabulario controlado (conjunto de definiciones y especializaciones).

```
Todos los estudiantes son personas.
Hay 2 tipos de personas: estudiantes y profesores.
```

Otros modelos lógicos

Lógicas descriptivas [DL: Description Logic]

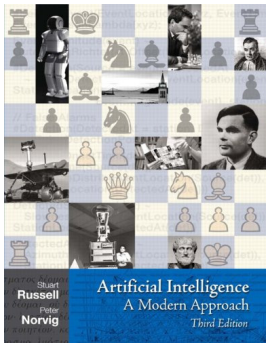
Base de conocimiento = ABox + TBox

Desde el punto de vista lógico, la distinción ABox/TBox no es esencial, pero resulta útil en la práctica

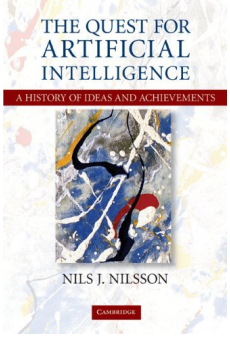
- para resolver problemas concretos
  (comprobación de instancias en ABox, clasificación en TBox)
- para modelar correctamente un dominio particular
  (términos/conceptos en TBox [clases] y manifestaciones particulares de esos conceptos en ABox [instancias]).

Bibliografía

- Stuart Russell & Peter Norvig:
  Artificial Intelligence:
  A Modern Approach
  Prentice-Hall, 3rd edition, 2009
  ISBN 0136042597



- Nils J. Nilsson
  The Quest for Artificial Intelligence
  Cambridge University Press, 2009
  ISBN 0521122937



Bibliografía

Bibliografía complementaria

- Elaine Rich & Kevin Knight: **Artificial Intelligence**. McGraw-Hill, 1991.
- Patrick Henry Winston: **Artificial Intelligence**. Addison-Wesley, 1992.
- Nils J. Nilsson: **Principles of Artificial Intelligence**. Morgan Kaufmann, 1986.
- Matt Ginsberg: **Essentials of Artificial Intelligence**. Morgan Kaufmann, 1993.
- Jack Minker (editor): **Logic-Based Artificial Intelligence**. Kluwer, 2000.