BLOQUE III: Tema 1

SINTAXIS DE LA LÓGICA DE PRIMER ORDEN

Lógica

Grado en Ingeniería Informática

Alessandra Gallinari

URJC

Contenido

Contenido

Contenido

Contenido

Introducción

La **lógica de predicados o de primer orden (LPO, L_1)** es una generalización de la lógica de proposiciones (LP, L_0). Introduciendo nuevos elementos del lenguaje, permite estudiar la estructura interna de los enunciados (sus propiedades, las relaciones entre objetos, etc.).

Introducción

La **lógica de predicados o de primer orden (LPO, L_1)** es una generalización de la lógica de proposiciones (LP, L_0). Introduciendo nuevos elementos del lenguaje, permite estudiar la estructura interna de los enunciados (sus propiedades, las relaciones entre objetos, etc.).

Ejemplo

Consideremos el siguiente razonamiento:

$$\frac{\begin{array}{l}\textit{Todos los hombres son mortales,}\\ \textit{Sócrates es un hombre,}\end{array}}{\textit{Sócrates es mortal.}}$$

Sean $p =$ *todos los hombres son mortales,* $q =$ *Sócrates es un hombre y* $r =$ *Sócrates es mortal. La formalización del razonamiento dado en lógica proposicional es* $p \wedge q \rightarrow r$ *y tiene el contraejemplo* $p : 1$, $q : 1$ *y* $r : 0$. *Por tanto, el razonamiento en estudio no es válido en lógica proposicional.*

Introducción

Es evidente que la conclusión de este ejemplo no satisface nuestra intuición.

Introducción

Es evidente que la conclusión de este ejemplo no satisface nuestra intuición.
Necesitamos extender la lógica proposicional a una lógica que permita dar una descripción más fina de la realidad, pudiendo distinguir los objetos o términos (por ejemplo, los hombres) de sus propiedades o predicados (por ejemplo, la propiedad de ser mortales).

Introducción

Es evidente que la conclusión de este ejemplo no satisface nuestra intuición.
Necesitamos extender la lógica proposicional a una lógica que permita dar una descripción más fina de la realidad, pudiendo distinguir los objetos o términos (por ejemplo, los hombres) de sus propiedades o predicados (por ejemplo, la propiedad de ser mortales).
La lógica de predicados (Gottob Frege, 1879) nos permite dar una descripción de la realidad más detallada. Por ejemplo, veremos que el razonamiento anterior se puede formalizar más precisamente como

$$\begin{array}{l} \forall x(H(x) \rightarrow M(x)), \\ H(s), \\ \hline M(s). \end{array}$$

Introducción

Como ya comentamos en varias ocasiones, la sintaxis es la definición axiomática de los elementos básicos del lenguaje y de las reglas que permiten obtener nuevas expresiones correctas a partir de aquellos, las expresiones bien construidas. El objetivo de este capítulo es el estudio de la sintaxis de la lógica de predicados.

Introducción

Como ya comentamos en varias ocasiones, la sintaxis es la definición axiomática de los elementos básicos del lenguaje y de las reglas que permiten obtener nuevas expresiones correctas a partir de aquellos, las expresiones bien construidas. El objetivo de este capítulo es el estudio de la sintaxis de la lógica de predicados.

Estudiamos una breve introducción a las nociones básicas de la teoría de conjuntos, de las relaciones y de las funciones. Estos conceptos son fundamentales para la lógica de predicados y se aconseja su repaso.

Contenido

Alfabeto del lenguaje formal de la lógica de predicados

Los elementos básicos del **alfabeto del la lógica de predicados** son:

Alfabeto del lenguaje formal de la lógica de predicados

Los elementos básicos del **alfabeto del la lógica de predicados** son:

- **Los símbolos de constantes:** se denotan $a, b, c, \ldots$ y representan objetos concretos. Las constantes son **individuos** o **elementos distinguidos** del **universo del discurso**, que es la colección de objetos sobre los cuales queremos razonar.

Alfabeto del lenguaje formal de la lógica de predicados

Los elementos básicos del **alfabeto del la lógica de predicados** son:

- **Los símbolos de constantes:** se denotan $a, b, c, \ldots$ y representan objetos concretos. Las constantes son **individuos** o **elementos distinguidos** del **universo del discurso**, que es la colección de objetos sobre los cuales queremos razonar.
- **Las variables:** se denotan $x, y, z, \ldots$ y sirven para representar objetos, cuyo dominio hay que especificar.
  Tomaremos conjuntos de variables V finitos o infinitos *numerables*.
  Recordamos que un conjunto V es infinito numerable si existe una función biyectiva entre V y el conjunto de los números naturales $\mathbb{N}$.

Alfabeto del lenguaje formal de la lógica de predicados

Los elementos básicos del **alfabeto del la lógica de predicados** son:

- **Los símbolos de constantes:** se denotan $a, b, c, \ldots$ y representan objetos concretos. Las constantes son **individuos** o **elementos distinguidos** del **universo del discurso**, que es la colección de objetos sobre los cuales queremos razonar.
- **Las variables:** se denotan $x, y, z, \ldots$ y sirven para representar objetos, cuyo dominio hay que especificar.
  Tomaremos conjuntos de variables V finitos o infinitos *numerables*.
  Recordamos que un conjunto V es infinito numerable si existe una función biyectiva entre V y el conjunto de los números naturales $\mathbb{N}$.

Ejemplo

En las expresiones "Pedro es un estudiante" y "x es un número primo," Pedro es una constante en el conjunto de la personas y x es una variable en el conjunto de los números enteros.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los conectivos lógicos:**

Alfabeto del lenguaje formal de la lógica de predicados

- **Los conectivos lógicos:**
- **constantes (de aridad 0):** $\top$ (verdadero) y $\perp$ (falso)

Alfabeto del lenguaje formal de la lógica de predicados

- **Los conectivos lógicos:**
- **constantes (de aridad 0):** $\top$ (verdadero) y $\bot$ (falso)
- **conectivos unarios:** $\neg$ (negación)

Alfabeto del lenguaje formal de la lógica de predicados

- **Los conectivos lógicos:**
- **constantes (de aridad 0):** $\top$ (verdadero) y $\bot$ (falso)
- **conectivos unarios:** $\neg$ (negación)

Ejemplo

Siendo "$E(a)$: Pedro es un estudiante," aplicando la negación se obtiene "$\neg(E(a))$: Pedro no es un estudiante." De forma similar, si "$P(x)$: x es un número primo," "$\neg(P(x))$: x" no es un número primo.

Alfabeto del lenguaje formal de la lógica de predicados

- **conectivos binarios:**

Alfabeto del lenguaje formal de la lógica de predicados

- **conectivos binarios:**
  $\wedge$ (y: la conjunción),

Alfabeto del lenguaje formal de la lógica de predicados

- **conectivos binarios:**
  $\wedge$ (y: la conjunción),
  $\vee$ (ó: la disyunción),

Alfabeto del lenguaje formal de la lógica de predicados

- **conectivos binarios:**
  $\wedge$ (y: la conjunción),
  $\vee$ (ó: la disyunción),
  $\rightarrow$ (la implicación o condicional),

Alfabeto del lenguaje formal de la lógica de predicados

- **conectivos binarios:**
  $\wedge$ (y: la conjunción),
  $\vee$ (ó: la disyunción),
  $\rightarrow$ (la implicación o condicional),
  $\leftrightarrow$ (la doble implicación o coimplicación).

Alfabeto del lenguaje formal de la lógica de predicados

- **conectivos binarios:**
  $\wedge$ (y: la conjunción),
  $\vee$ (ó: la disyunción),
  $\rightarrow$ (la implicación o condicional),
  $\leftrightarrow$ (la doble implicación o coimplicación).

NOTACIÓN: El símbolo $\circ$ se usará para representar un conectivo binario cualquiera.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplos

1) La frase "Sócrates es un filósofo, sin embargo no es un deportista" se escribe como $(F(s) \wedge \neg(D(s)))$, *donde "*$F(x)$: x *es un filósofo," "*$D(x)$: x *es un deportista" y* x *es una variable en el conjunto de las personas.*

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplos

1) La frase "Sócrates es un filósofo, sin embargo no es un deportista" se escribe como $(F(s) \wedge \neg(D(s)))$*, donde "*$F(x) : x$ *es un filósofo," "*$D(x) : x$ *es un deportista" y* x *es una variable en el conjunto de las personas.*
2) La frase "Sócrates es un filósofo o Sócrates es un deportista" se escribe como $(F(s) \vee D(s))$.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplos

1) La frase "Sócrates es un filósofo, sin embargo no es un deportista" se escribe como $(F(s) \wedge \neg(D(s)))$*, donde "*$F(x) : x$ *es un filósofo," "*$D(x) : x$ *es un deportista" y* x *es una variable en el conjunto de las personas.*

2) La frase "Sócrates es un filósofo o Sócrates es un deportista" se escribe como $(F(s) \vee D(s))$.

3) La frase "La luna es de papel si y sólo si Carlos lee muchos libros" se escribe como $(P(l) \leftrightarrow L(c))$*, donde "*$P(x) : x$ *es de papel," "*$L(y) : y$ *lee muchos libros,"* x *es una variable en el conjunto de las cosas y* y *es una variable en el conjunto de las personas.*

Alfabeto del lenguaje formal de la lógica de predicados

- **Igualdad:**
  Usaremos el símbolo de igualdad "$=$" para expresiones del tipo $4-1=3$ o $mcm(2,3)=6$.

Alfabeto del lenguaje formal de la lógica de predicados

- **Igualdad:**
  Usaremos el símbolo de igualdad "$=$" para expresiones del tipo $4 - 1 = 3$ o $mcm(2, 3) = 6$.
  Por tanto, estamos considerando **la lógica de predicados con igualdad**.

Alfabeto del lenguaje formal de la lógica de predicados

- **Igualdad:**
  Usaremos el símbolo de igualdad "$=$" para expresiones del tipo $4-1=3$ o $mcm(2,3)=6$.
  Por tanto, estamos considerando **la lógica de predicados con igualdad**.

Ejemplo

El comando "IF $x>0$ THEN $y=\sqrt{x}$" se escribe como $(P(x)\rightarrow(y=f(x)))$, donde "$P(x): x$ es positivo," "$f(x):\sqrt{x}$" y x y y son variables en el conjunto de los números reales.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los cuantificadores:**

Alfabeto del lenguaje formal de la lógica de predicados

- **Los cuantificadores:**
  1. $\forall$: **cuantificador universal** (para todo). El cuantificador universal permite referirse a **todos** los individuos del universo del discurso.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los cuantificadores:**
    1. $\forall$: **cuantificador universal** (para todo). El cuantificador universal permite referirse a **todos** los individuos del universo del discurso.
    2. $\exists$: **cuantificador existencial** (existe). El cuantificador existencial permite referirse a **algunos** de los individuos del universo del discurso.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los cuantificadores:**
  1. $\forall$: **cuantificador universal** (para todo). El cuantificador universal permite referirse a **todos** los individuos del universo del discurso.
  2. $\exists$: **cuantificador existencial** (existe). El cuantificador existencial permite referirse a **algunos** de los individuos del universo del discurso.

Ejemplos

1) La frase "Todo número primo y mayor que 2 es impar" se escribe como $(\forall x((P(x) \wedge Q(x)) \rightarrow R(x)))$, *donde "*$P(x) : x$ *es primo," "*$Q(x) : x$ *es mayor que 2," "*$R(x) : x$ *es impar" y* x *es un elemento cualquiera del conjunto de los números enteros.*

Alfabeto del lenguaje formal de la lógica de predicados

- **Los cuantificadores:**
  1. $\forall$: **cuantificador universal** (para todo). El cuantificador universal permite referirse a **todos** los individuos del universo del discurso.
  2. $\exists$: **cuantificador existencial** (existe). El cuantificador existencial permite referirse a **algunos** de los individuos del universo del discurso.

Ejemplos

1) La frase "Todo número primo y mayor que 2 es impar" se escribe como $(\forall x((P(x) \wedge Q(x)) \rightarrow R(x)))$, *donde "*$P(x) : x$ *es primo," "*$Q(x) : x$ *es mayor que 2," "*$R(x) : x$ *es impar" y* x *es un elemento cualquiera del conjunto de los números enteros.*

2) La frase "Todo hombre es mortal y hay hombres que no son filósofos" se escribe como $((\forall x(P(x) \rightarrow Q(x))) \wedge (\exists x(P(x) \wedge (\neg(R(x))))))$, *donde "*$P(x) : x$ *es un hombre," "*$Q(x) : x$ *es mortal," "*$R(x) : x$ *es filósofo" y* x *es un elemento cualquiera del conjunto de los seres.*

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de puntuación (o símbolos auxiliares):** paréntesis abiertos y cerrados, comas.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de puntuación (o símbolos auxiliares):** paréntesis abiertos y cerrados, comas.
- **Los símbolos de predicado:** se denotan $P, Q, R, \ldots$.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de puntuación (o símbolos auxiliares):** paréntesis abiertos y cerrados, comas.
- **Los símbolos de predicado:** se denotan $P, Q, R, \ldots$.
  Todo predicado tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** del predicado. En ocasiones se especificará la aridad n de un predicado P por medio del símbolo P^n.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de puntuación (o símbolos auxiliares):** paréntesis abiertos y cerrados, comas.
- **Los símbolos de predicado:** se denotan $P, Q, R, \dots$.
  Todo predicado tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** del predicado. En ocasiones se especificará la aridad n de un predicado P por medio del símbolo P^n.
  1. **Predicados constantes,** $n = 0$: representan **proposiciones atómicas**. Para representar las proposiciones atómicas se suelen usar los símbolos p, q, r, s, $t, \dots$

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de puntuación (o símbolos auxiliares):** paréntesis abiertos y cerrados, comas.
- **Los símbolos de predicado:** se denotan $P, Q, R, \ldots$.
  Todo predicado tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** del predicado. En ocasiones se especificará la aridad n de un predicado P por medio del símbolo P^n.
  1. **Predicados constantes,** $n = 0$: representan **proposiciones atómicas**. Para representar las proposiciones atómicas se suelen usar los símbolos p, q, r, s, $t, \ldots$
  2. **Predicados monádicos,** $n = 1$: representan **propiedades** de objetos.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de puntuación (o símbolos auxiliares):** paréntesis abiertos y cerrados, comas.
- **Los símbolos de predicado:** se denotan $P, Q, R, \ldots$.
  Todo predicado tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** del predicado. En ocasiones se especificará la aridad n de un predicado P por medio del símbolo P^n.
  1. **Predicados constantes,** $n = 0$: representan **proposiciones atómicas**. Para representar las proposiciones atómicas se suelen usar los símbolos p, q, r, s, $t, \ldots$
  2. **Predicados monádicos,** $n = 1$: representan **propiedades** de objetos.
  3. **Predicados poliádicos,** $n > 1$: representan **relaciones** entre objetos.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de puntuación (o símbolos auxiliares):** paréntesis abiertos y cerrados, comas.
- **Los símbolos de predicado:** se denotan $P, Q, R, \ldots$.
  Todo predicado tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** del predicado. En ocasiones se especificará la aridad n de un predicado P por medio del símbolo P^n.
  1. **Predicados constantes,** $n = 0$: representan **proposiciones atómicas**. Para representar las proposiciones atómicas se suelen usar los símbolos p, q, r, s, $t, \ldots$
  2. **Predicados monádicos,** $n = 1$: representan **propiedades** de objetos.
  3. **Predicados poliádicos,** $n > 1$: representan **relaciones** entre objetos.

  Los predicados poliádicos de la lógica de primer orden son relaciones sobre conjuntos. Así, por ejemplo, todo predicado binario es una relación binaria R entre dos conjuntos A y B, es decir, $R \subseteq A \times B$.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de puntuación (o símbolos auxiliares):** paréntesis abiertos y cerrados, comas.
- **Los símbolos de predicado:** se denotan $P, Q, R, \ldots$.
  Todo predicado tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** del predicado. En ocasiones se especificará la aridad n de un predicado P por medio del símbolo P^n.
  1. **Predicados constantes,** $n = 0$: representan **proposiciones atómicas**. Para representar las proposiciones atómicas se suelen usar los símbolos p, q, r, s, $t, \ldots$
  2. **Predicados monádicos,** $n = 1$: representan **propiedades** de objetos.
  3. **Predicados poliádicos,** $n > 1$: representan **relaciones** entre objetos.

  Los predicados poliádicos de la lógica de primer orden son relaciones sobre conjuntos. Así, por ejemplo, todo predicado binario es una relación binaria R entre dos conjuntos A y B, es decir, $R \subseteq A \times B$. Un predicado monádico asocia a cada objeto de un dominio una propiedad.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de puntuación (o símbolos auxiliares):** paréntesis abiertos y cerrados, comas.
- **Los símbolos de predicado:** se denotan $P, Q, R, \ldots$.
  Todo predicado tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** del predicado. En ocasiones se especificará la aridad n de un predicado P por medio del símbolo P^n.
  1. **Predicados constantes,** $n = 0$: representan **proposiciones atómicas**. Para representar las proposiciones atómicas se suelen usar los símbolos p, q, r, s, $t, \ldots$
  2. **Predicados monádicos,** $n = 1$: representan **propiedades** de objetos.
  3. **Predicados poliádicos,** $n > 1$: representan **relaciones** entre objetos.

  Los predicados poliádicos de la lógica de primer orden son relaciones sobre conjuntos. Así, por ejemplo, todo predicado binario es una relación binaria R entre dos conjuntos A y B, es decir, $R \subseteq A \times B$. Un predicado monádico asocia a cada objeto de un dominio una propiedad.

  Nota: Un predicado es tal que al sustituir todas sus símbolos de variables por símbolos de constantes se obtiene una proposición atómica.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Los siguientes son ejemplos de predicados:

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Los siguientes son ejemplos de predicados:
a) "p : hoy llueve" es un predicado constante, una proposición atómica.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Los siguientes son ejemplos de predicados:
a) "p : hoy llueve" es un predicado constante, una proposición atómica.
b) "$P(x)$: la raíz cuadrada de x es irracional" (predicado monádico), siendo x un cualquier número real.
Al sustituir x por un símbolo de constante, por ejemplo 2, se obtiene la proposición "$P(2)$: la raíz cuadrada de 2 es irracional."

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Los siguientes son ejemplos de predicados:
a) "p : hoy llueve" es un predicado constante, una proposición atómica.
b) "$P(x)$: la raíz cuadrada de x es irracional" (predicado monádico), siendo x un cualquier número real.
Al sustituir x por un símbolo de constante, por ejemplo 2, se obtiene la proposición "$P(2)$: la raíz cuadrada de 2 es irracional."
c) $P(x, y)$: x es un hermano de y (predicado binario), siendo el dominio el conjunto de las personas.
En particular, "$P(l, j)$: $Luis$ es un hermano de $Jose$" es una fórmula proposicional.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Los siguientes son ejemplos de predicados:
a) "p : hoy llueve" es un predicado constante, una proposición atómica.
b) "$P(x)$: la raíz cuadrada de x es irracional" (predicado monádico), siendo x un cualquier número real.
Al sustituir x por un símbolo de constante, por ejemplo 2, se obtiene la proposición "$P(2)$: la raíz cuadrada de 2 es irracional."
c) $P(x, y)$: x es un hermano de y (predicado binario), siendo el dominio el conjunto de las personas.
En particular, "$P(l, j)$: $Luis$ es un hermano de $Jose$" es una fórmula proposicional.
d) $P(x, y, z)$: x prefiere y a z (predicado ternario), siendo el dominio de las variable x el conjunto de las personas y el dominio de las variables y y z el conjunto de los países.
En particular, "$P(l, f, i)$: $Luis$ prefiere $Francia$ a $Italia$" es una fórmula proposicional.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Como vimos, los predicados permiten definir conjuntos.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Como vimos, los predicados permiten definir conjuntos.
Por ejemplo, en el dominio de los puntos del plano real $\mathbb{R} \times \mathbb{R}$*, el conjunto* S *de las soluciones del sistema de dos ecuaciones*

$$\begin{cases} 3x + 2y & = 1 \\ x - 4y & = 0 \end{cases}$$

se puede definir por medio de los predicados "$S_1(z)$ *: el punto* $z = (x, y)$ *es tal que* $3x + 2y = 1$*" y "*$S_2(z)$ *: el punto* $z = (x, y)$ *es tal que* $x - 4y = 0$*."*

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Como vimos, los predicados permiten definir conjuntos.
Por ejemplo, en el dominio de los puntos del plano real $\mathbb{R} \times \mathbb{R}$, *el conjunto* S *de las soluciones del sistema de dos ecuaciones*

$$\begin{cases} 3x + 2y & = 1 \\ x - 4y & = 0 \end{cases}$$

se puede definir por medio de los predicados "$S_1(z)$: *el punto* $z = (x, y)$ *es tal que* $3x + 2y = 1$*" y "*$S_2(z)$: *el punto* $z = (x, y)$ *es tal que* $x - 4y = 0$.*"*
En el siguiente tema, dedicado a la semántica de la lógica de primer orden, veremos como asignar valores de verdad a las expresiones bien construidas.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Como vimos, los predicados permiten definir conjuntos.
Por ejemplo, en el dominio de los puntos del plano real $\mathbb{R} \times \mathbb{R}$*, el conjunto* S *de las soluciones del sistema de dos ecuaciones*

$$\begin{cases} 3x + 2y & = 1 \\ x - 4y & = 0 \end{cases}$$

se puede definir por medio de los predicados "$S_1(z)$ *: el punto* $z = (x, y)$ *es tal que* $3x + 2y = 1$*" y "*$S_2(z)$ *: el punto* $z = (x, y)$ *es tal que* $x - 4y = 0$*."*
En el siguiente tema, dedicado a la semántica de la lógica de primer orden, veremos como asignar valores de verdad a las expresiones bien construidas.
Para nuestro ejemplo, podremos afirmar que los elementos del conjunto S *son los puntos del plano* z *tales que* $S_1(z) \wedge S_2(z)$ *es verdadera.*

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de función:** se denotan $f, g, h, \ldots$.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de función:** se denotan $f, g, h, \ldots$.
  Toda función tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** de la función.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de función:** se denotan $f, g, h, \ldots$.
  Toda función tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** de la función.
  En ocasiones se especificará la aridad n de una función f por medio del símbolo f^n.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de función:** se denotan $f, g, h, \ldots$.
  Toda función tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** de la función.
  En ocasiones se especificará la aridad n de una función f por medio del símbolo f^n.
  Nota: Las funciones de la lógica de primer orden son funciones.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de función:** se denotan $f, g, h, \ldots$.
  Toda función tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** de la función.
  En ocasiones se especificará la aridad n de una función f por medio del símbolo f^n.
  Nota: Las funciones de la lógica de primer orden son funciones.
  1. **Funciones constantes,** $n = 0$: son símbolos de constantes.
     Para representar las funciones constantes se suelen usar los símbolos a, b, $c, \ldots$

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de función:** se denotan $f, g, h, \ldots$.
  Toda función tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** de la función.
  En ocasiones se especificará la aridad n de una función f por medio del símbolo f^n.
  Nota: Las funciones de la lógica de primer orden son funciones.
  1. **Funciones constantes,** $n = 0$: son símbolos de constantes.
     Para representar las funciones constantes se suelen usar los símbolos a, b, $c, \ldots$
  2. **Funciones monádicas,** $n = 1$: representan un objeto en función de otro.

Alfabeto del lenguaje formal de la lógica de predicados

- **Los símbolos de función:** se denotan $f, g, h, \ldots$.
  Toda función tiene un número $n \in \mathbb{N} \cup \{0\}$ de argumentos. El número n es la **aridad** de la función.
  En ocasiones se especificará la aridad n de una función f por medio del símbolo f^n.
  Nota: Las funciones de la lógica de primer orden son funciones.
  1. **Funciones constantes,** $n = 0$: son símbolos de constantes.
     Para representar las funciones constantes se suelen usar los símbolos a, b, c, ...
  2. **Funciones monádicas,** $n = 1$: representan un objeto en función de otro.
  3. **Funciones poliádicas,** $n > 1$: representan un objeto en función de n otros objetos.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Los siguientes son ejemplos de funciones:

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Los siguientes son ejemplos de funciones:
a) $f(x) : \sqrt{x}$ *es la función monádica raíz cuadrada de un numero real no negativo,* $f \subseteq [0, \infty) \times \mathbb{R}$.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Los siguientes son ejemplos de funciones:
a) $f(x) : \sqrt{x}$ *es la función monádica raíz cuadrada de un numero real no negativo,* $f \subseteq [0, \infty) \times \mathbb{R}$.
b) $f(x, y) : x + y$ *es la función binaria suma de dos números reales,* $f \subseteq \mathbb{R} \times \mathbb{R} \times \mathbb{R}$.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

Los siguientes son ejemplos de funciones:
a) $f(x) : \sqrt{x}$ *es la función monádica raíz cuadrada de un numero real no negativo,* $f \subseteq [0, \infty) \times \mathbb{R}$.
b) $f(x, y) : x + y$ *es la función binaria suma de dos números reales,* $f \subseteq \mathbb{R} \times \mathbb{R} \times \mathbb{R}$.
c) $f(x, g(y, z), a) : x(y + z) - a$ *es la función ternaria,* $f \subseteq \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R}$, *que multiplica las primeras dos variables reales y resta al producto obtenido la constante real* a.
g *es la función binaria,* $g \subseteq \mathbb{R} \times \mathbb{R} \times \mathbb{R}$, *suma de dos números reales.*

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

El siguiente es un ejemplo de un predicado que no es una función:

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

El siguiente es un ejemplo de un predicado que no es una función:
En el conjunto de las personas, "$P(x, y) = x$ es la madre de y" es un predicado (una relación binaria) que no es una función, ya que su dominio no es todo el conjunto de las personas y una madre puede tener más que un hijo.

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

El siguiente es un ejemplo de un predicado que no es una función:
En el conjunto de las personas, "$P(x, y) = x$ es la madre de y" es un predicado (una relación binaria) que no es una función, ya que su dominio no es todo el conjunto de las personas y una madre puede tener más que un hijo.

Observación

De los ejemplos anteriores se deduce que, si n es un número natural, **toda función $n-$aria se puede escribir como un predicado $(n+1)$-nario.** *Sin embargo, en general, un predicado no se puede escribir como una función.*

Alfabeto del lenguaje formal de la lógica de predicados

Ejemplo

El siguiente es un ejemplo de un predicado que no es una función:
En el conjunto de las personas, "$P(x, y) = x$ es la madre de y" es un predicado (una relación binaria) que no es una función, ya que su dominio no es todo el conjunto de las personas y una madre puede tener más que un hijo.

Observación

De los ejemplos anteriores se deduce que, si n es un número natural, **toda función $n-$aria se puede escribir como un predicado $(n+1)$-nario.** *Sin embargo, en general, un predicado no se puede escribir como una función.*
Veremos que las funciones sirven para simplificar la estructura de la fórmulas de la lógica de primer orden.

Contenido

Definición recursiva de términos y fórmulas

Como en la lógica proposicional, los elementos básicos de un lenguaje y las reglas de formación permiten definir cadenas finitas de símbolos arbitrarias (**palabras**).

Definición recursiva de términos y fórmulas

Como en la lógica proposicional, los elementos básicos de un lenguaje y las reglas de formación permiten definir cadenas finitas de símbolos arbitrarias (**palabras**).
En la lógica proposicional la palabras que son expresiones bien construidas son las fórmulas. En la lógica de predicados las expresiones bien construidas pueden ser de dos tipos: **términos** (que representan objetos) y **fórmulas** (que expresan hechos relativos a objetos).

Definición recursiva de términos y fórmulas

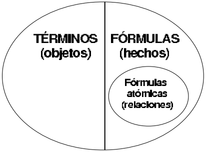


Figura: Expresiones bien construidas

Los términos y las fórmulas son las **expresiones bien construidas** de la lógica de primer orden (ver figura 1).

Definición recursiva de términos y fórmulas

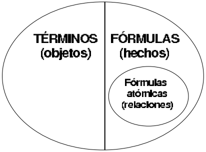


Figura: Expresiones bien construidas

Los términos y las fórmulas son las **expresiones bien construidas** de la lógica de primer orden (ver figura 1).
Los términos y las fórmulas de la lógica de predicados se obtienen a partir del alfabeto dado sólo por medio de las reglas de formación que vamos a definir a continuación.

Definición recursiva de términos y fórmulas

Como en la lógica proposicional, la definición de las expresiones bien construidas de la lógica de primer orden es recursiva y, como veremos, siguen valiendo los principios de unicidad de estructura, de inducción y de recursión estructural para términos y fórmulas.

Definición recursiva de términos y fórmulas

Como en la lógica proposicional, la definición de las expresiones bien construidas de la lógica de primer orden es recursiva y, como veremos, siguen valiendo los principios de unicidad de estructura, de inducción y de recursión estructural para términos y fórmulas.

- **Términos**

Definición

(Definición recursiva de los términos)

Definición recursiva de términos y fórmulas

Como en la lógica proposicional, la definición de las expresiones bien construidas de la lógica de primer orden es recursiva y, como veremos, siguen valiendo los principios de unicidad de estructura, de inducción y de recursión estructural para términos y fórmulas.

- **Términos**

Definición

(Definición recursiva de los términos)

1. *(TAt): Todo símbolo de variable y todo símbolo de constante es un término.*

   Las variables y las constantes forman el conjunto de los **términos atómicos***.*

Definición recursiva de términos y fórmulas

Como en la lógica proposicional, la definición de las expresiones bien construidas de la lógica de primer orden es recursiva y, como veremos, siguen valiendo los principios de unicidad de estructura, de inducción y de recursión estructural para términos y fórmulas.

- **Términos**

Definición

(Definición recursiva de los términos)

1. *(TAt): Todo símbolo de variable y todo símbolo de constante es un término.*

   Las variables y las constantes forman el conjunto de los **términos atómicos**.

2. *(TF): Si* f *es un símbolo de función de aridad* $n > 0$ *y* $t_1, t_2, \ldots, t_n$ *son términos, entonces* $f(t_1, t_2, \ldots, t_n)$ *es un término.*

   Estos términos se llaman **términos compuestos**.

Definición recursiva de términos y fórmulas

Como en la lógica proposicional, la definición de las expresiones bien construidas de la lógica de primer orden es recursiva y, como veremos, siguen valiendo los principios de unicidad de estructura, de inducción y de recursión estructural para términos y fórmulas.

- **Términos**

Definición

(Definición recursiva de los términos)

1. *(TAt): Todo símbolo de variable y todo símbolo de constante es un término.*

   Las variables y las constantes forman el conjunto de los **términos atómicos**.

2. *(TF): Si* f *es un símbolo de función de aridad* $n > 0$ *y* $t_1, t_2, \ldots, t_n$ *son términos, entonces* $f(t_1, t_2, \ldots, t_n)$ *es un término.*

   Estos términos se llaman **términos compuestos**.

3. *Si una palabra no se obtiene mediante las dos reglas anteriores, entonces no es un término.*

Definición recursiva de términos y fórmulas

Ejemplo

Los siguientes son ejemplos de términos:

$$x, a, f(x), g(x,y), g(x, f(x)),$$

donde x *es una variable,* a *es una constante,* f *es una función monádica y* g *es una función binaria.*
Los primeros dos términos de la lista son atómicos y los restantes son compuestos.

Definición recursiva de términos y fórmulas

Ejemplo

Los siguientes son ejemplos de términos:

$$x, a, f(x), g(x,y), g(x, f(x)),$$

donde x es una variable, a es una constante, f es una función monádica y g es una función binaria.
Los primeros dos términos de la lista son atómicos y los restantes son compuestos.

- **Fórmulas**

Definición

Una **fórmula atómica** *es cualquier expresión de la forma $P(t_1, t_2, \ldots, t_n)$, donde P es un símbolo de predicado con aridad $n > 0$ y $t_1, t_2, \ldots, t_n$ son términos. Las proposiciones atómicas (los predicados constantes), los conectivos lógicos constantes $\top$ y $\bot$ y la igualdad entre términos, $s = t$, también se consideran fórmulas atómicas.*

Definición recursiva de términos y fórmulas

Ejemplo

Los siguientes son ejemplos de fórmulas atómicas: $\top$, p, $R(f(x), y)$, *donde* p *es una proposición atómica,* f *es una función monádica y* R *es un predicado binario.*

Definición recursiva de términos y fórmulas

Ejemplo

Los siguientes son ejemplos de fórmulas atómicas: $\top$, p, $R(f(x), y)$, *donde* p *es una proposición atómica,* f *es una función monádica y* R *es un predicado binario.*

Definición

(Definición recursiva de las fórmulas)

Definición recursiva de términos y fórmulas

Ejemplo

Los siguientes son ejemplos de fórmulas atómicas: $\top$, p, $R(f(x), y)$, *donde* p *es una proposición atómica,* f *es una función monádica y* R *es un predicado binario.*

Definición

(Definición recursiva de las fórmulas)

1. *(FAt): Toda fórmula atómica es una fórmula.*

Definición recursiva de términos y fórmulas

Ejemplo

Los siguientes son ejemplos de fórmulas atómicas: $\top$, p, $R(f(x), y)$, *donde* p *es una proposición atómica,* f *es una función monádica y* R *es un predicado binario.*

Definición

(Definición recursiva de las fórmulas)

1. *(FAt): Toda fórmula atómica es una fórmula.*
2. *(F$\neg$): Si* φ *es una fórmula, entonces* $\neg\varphi$ *es una fórmula.*

Definición recursiva de términos y fórmulas

Ejemplo

Los siguientes son ejemplos de fórmulas atómicas: $\top$, p, $R(f(x), y)$, *donde* p *es una proposición atómica,* f *es una función monádica y* R *es un predicado binario.*

Definición

(Definición recursiva de las fórmulas)

1. *(FAt): Toda fórmula atómica es una fórmula.*
2. *(F¬): Si* φ *es una fórmula, entonces* $\neg\varphi$ *es una fórmula.*
3. *(F∘): Si* φ *y* ψ *son dos fórmulas entonces* $(\varphi \circ \psi)$ *es una fórmula.*

Definición recursiva de términos y fórmulas

Ejemplo

Los siguientes son ejemplos de fórmulas atómicas: $\top$, p, $R(f(x), y)$, *donde* p *es una proposición atómica,* f *es una función monádica y* R *es un predicado binario.*

Definición

(Definición recursiva de las fórmulas)

1. *(FAt): Toda fórmula atómica es una fórmula.*
2. *(F$\neg$): Si* φ *es una fórmula, entonces* $\neg\varphi$ *es una fórmula.*
3. *(F$\circ$): Si* φ *y* ψ *son dos fórmulas entonces* $(\varphi \circ \psi)$ *es una fórmula.*
4. *(F$\forall\exists$): Si* φ *es una fórmula y* x *es un símbolo de variable, entonces* $\exists x\varphi$ *y* $\forall x\varphi$ *son fórmulas.*

Definición recursiva de términos y fórmulas

Ejemplo

Los siguientes son ejemplos de fórmulas atómicas: $\top$, p, $R(f(x), y)$, *donde* p *es una proposición atómica,* f *es una función monádica y* R *es un predicado binario.*

Definición

(Definición recursiva de las fórmulas)

1. *(FAt): Toda fórmula atómica es una fórmula.*
2. *(F¬): Si* φ *es una fórmula, entonces* $\neg\varphi$ *es una fórmula.*
3. *(F∘): Si* φ *y* ψ *son dos fórmulas entonces* $(\varphi \circ \psi)$ *es una fórmula.*
4. *(F∀∃): Si* φ *es una fórmula y* x *es un símbolo de variable, entonces* $\exists x\varphi$ *y* $\forall x\varphi$ *son fórmulas.*
5. *Si una palabra no se obtiene mediante las cuatro reglas anteriores, entonces no es una fórmula.*

Definición recursiva de términos y fórmulas

NOTACIÓN: Para representar las fórmulas se suelen usar las letras del alfabeto griego $\varphi, \psi, \chi, \ldots$

Definición recursiva de términos y fórmulas

NOTACIÓN: Para representar las fórmulas se suelen usar las letras del alfabeto griego $\varphi, \psi, \chi, \dots$

Ejemplos

1) Las fórmulas atómicas del ejemplo anterior son fórmulas.

Definición recursiva de términos y fórmulas

NOTACIÓN: Para representar las fórmulas se suelen usar las letras del alfabeto griego $\varphi, \psi, \chi, \ldots$

Ejemplos

1) Las fórmulas atómicas del ejemplo anterior son fórmulas.
2) La expresión

$$(\forall x \exists y (R(x, f(y)) \wedge (\neg (f(x) = s)))$$

es una fórmula ya que:
1. $R(x, f(y))$ *y* $(f(x) = s)$ *son fórmulas atómicas (aplicando (FAt)),*

Definición recursiva de términos y fórmulas

NOTACIÓN: Para representar las fórmulas se suelen usar las letras del alfabeto griego $\varphi, \psi, \chi, \ldots$

Ejemplos

1) Las fórmulas atómicas del ejemplo anterior son fórmulas.
2) La expresión

$$(\forall x \exists y (R(x, f(y)) \wedge (\neg(f(x) = s))))$$

es una fórmula ya que:
1. $R(x, f(y))$ *y* $(f(x) = s)$ *son fórmulas atómicas (aplicando (FAt)),*
2. $(\neg(f(x) = s))$ *es una fórmula (aplicando (F¬)),*

Definición recursiva de términos y fórmulas

NOTACIÓN: Para representar las fórmulas se suelen usar las letras del alfabeto griego $\varphi, \psi, \chi, \dots$

Ejemplos

1) Las fórmulas atómicas del ejemplo anterior son fórmulas.
2) La expresión

$$(\forall x \exists y (R(x, f(y)) \wedge (\neg(f(x) = s)))$$

es una fórmula ya que:
1. $R(x, f(y))$ *y* $(f(x) = s)$ *son fórmulas atómicas (aplicando (FAt)),*
2. $(\neg(f(x) = s))$ *es una fórmula (aplicando (F¬)),*
3. $(R(x, f(y)) \wedge (\neg(f(x) = s)))$ *es una fórmula (aplicando* $(F\circ)$*),*

Definición recursiva de términos y fórmulas

NOTACIÓN: Para representar las fórmulas se suelen usar las letras del alfabeto griego $\varphi, \psi, \chi, \ldots$

Ejemplos

1) Las fórmulas atómicas del ejemplo anterior son fórmulas.
2) La expresión

$$(\forall x \exists y (R(x, f(y)) \wedge (\neg (f(x) = s)))$$

es una fórmula ya que:
1. $R(x, f(y))$ *y* $(f(x) = s)$ *son fórmulas atómicas (aplicando (FAt)),*
2. $(\neg (f(x) = s))$ *es una fórmula (aplicando (F¬)),*
3. $(R(x, f(y)) \wedge (\neg (f(x) = s)))$ *es una fórmula (aplicando* $(F\circ)$*),*
4. $\exists y (R(x, f(y)) \wedge (\neg (f(x) = s)))$ *es una fórmula (aplicando* $(F\forall\exists)$*),*

Definición recursiva de términos y fórmulas

NOTACIÓN: Para representar las fórmulas se suelen usar las letras del alfabeto griego $\varphi, \psi, \chi, \ldots$

Ejemplos

1) Las fórmulas atómicas del ejemplo anterior son fórmulas.
2) La expresión

$$(\forall x \exists y (R(x, f(y)) \wedge (\neg(f(x) = s)))$$

es una fórmula ya que:
1. $R(x, f(y))$ *y* $(f(x) = s)$ *son fórmulas atómicas (aplicando (FAt)),*
2. $(\neg(f(x) = s))$ *es una fórmula (aplicando (F¬)),*
3. $(R(x, f(y)) \wedge (\neg(f(x) = s)))$ *es una fórmula (aplicando* $(F\circ)$*),*
4. $\exists y (R(x, f(y)) \wedge (\neg(f(x) = s)))$ *es una fórmula (aplicando* $(F\forall\exists)$*),*
5. $(\forall x \exists y (R(x, f(y)) \wedge (\neg(f(x) = s))))$ *es una fórmula (aplicando* $(F\forall\exists)$*).*

Variables libres y variables ligadas

Las fórmulas se clasifican en **fórmulas abiertas** y **fórmulas cerradas**, dependiendo de ciertas características de sus variables.

Variables libres y variables ligadas

Las fórmulas se clasifican en **fórmulas abiertas** y **fórmulas cerradas**, dependiendo de ciertas características de sus variables.

Definición

Fórmulas abiertas y fórmulas cerradas

- *Se dice que la ocurrencia de una variable x en una fórmula φ es* **ligada** *(o que la variable es ligada) si está afectada por algún cuantificador. En caso de que una variable no aparezca ligada diremos que su ocurrencia es* **libre** *(la variable es libre).*

Variables libres y variables ligadas

Las fórmulas se clasifican en **fórmulas abiertas** y **fórmulas cerradas**, dependiendo de ciertas características de sus variables.

Definición

Fórmulas abiertas y fórmulas cerradas

- *Se dice que la ocurrencia de una variable x en una fórmula φ es* **ligada** *(o que la variable es ligada) si está afectada por algún cuantificador. En caso de que una variable no aparezca ligada diremos que su ocurrencia es* **libre** *(la variable es libre).*
- *Se dice que una fórmula φ es* **abierta** *si tiene alguna ocurrencia de variable libre. Se dice que una fórmula φ es* **cerrada** *(o que es una sentencia) si no tiene ninguna ocurrencia de variable libre.*

Variables libres y variables ligadas

Ejemplos

1) En la fórmula $\forall x P(x, y)$ *la variable* x *aparece ligada y la variable* y *aparece libre. La fórmula es abierta.*

Variables libres y variables ligadas

Ejemplos

1) En la fórmula $\forall x P(x, y)$ la variable x aparece ligada y la variable y aparece libre. La fórmula es abierta.
2) En la fórmula $\exists x((P(x) \wedge Q(x)) \vee (P(x) \wedge Q(x)))$ la variable x aparece ligada en las dos componentes de la disyunción, ya que el cuantificador existencial la afecta en los dos casos. La fórmula es cerrada.

Variables libres y variables ligadas

Ejemplos

1) En la fórmula $\forall x P(x,y)$ *la variable* x *aparece ligada y la variable* y *aparece libre. La fórmula es abierta.*
2) En la fórmula $\exists x((P(x) \wedge Q(x)) \vee (P(x) \wedge Q(x)))$ *la variable* x *aparece ligada en las dos componentes de la disyunción, ya que el cuantificador existencial la afecta en los dos casos. La fórmula es cerrada.*
3) En la fórmula $(\exists x(P(x) \wedge Q(x)) \vee (P(x) \wedge Q(x)))$ *la variable* x *aparece ligada en la primera componente de la disyunción y aparece libre en la segunda. En este caso el cuantificador existencial afecta* x *sólo en la primera componente de la disyunción y la fórmula es abierta.*

Variables libres y variables ligadas

Ejemplos

1) En la fórmula $\forall x P(x, y)$ *la variable* x *aparece ligada y la variable* y *aparece libre. La fórmula es abierta.*
2) En la fórmula $\exists x((P(x) \wedge Q(x)) \vee (P(x) \wedge Q(x)))$ *la variable* x *aparece ligada en las dos componentes de la disyunción, ya que el cuantificador existencial la afecta en los dos casos. La fórmula es cerrada.*
3) En la fórmula $(\exists x(P(x) \wedge Q(x)) \vee (P(x) \wedge Q(x)))$ *la variable* x *aparece ligada en la primera componente de la disyunción y aparece libre en la segunda. En este caso el cuantificador existencial afecta* x *sólo en la primera componente de la disyunción y la fórmula es abierta.*
Notar que esta última fórmula se puede reescribir como

$$(\exists x(P(x) \wedge Q(x)) \vee (P(y) \wedge Q(y))).$$

El principio de inducción estructural para términos

Como en el caso de la lógica de proposiciones, para poder estudiar las **propiedades** de las expresiones bien construidas de la lógica de primer orden una de las técnicas más adecuadas es el **principio de inducción estructural**.

El principio de inducción estructural para términos

Como en el caso de la lógica de proposiciones, para poder estudiar las **propiedades** de las expresiones bien construidas de la lógica de primer orden una de las técnicas más adecuadas es el **principio de inducción estructural**.

Principio de inducción estructural para términos:

Sea $\mathbb{P}$ una cierta propiedad tal que:

1. **Base de inducción (TAt):** Todos los términos atómicos cumplen la propiedad $\mathbb{P}$.

El principio de inducción estructural para términos

Como en el caso de la lógica de proposiciones, para poder estudiar las **propiedades** de las expresiones bien construidas de la lógica de primer orden una de las técnicas más adecuadas es el **principio de inducción estructural**.

Principio de inducción estructural para términos:

Sea $\mathbb{P}$ una cierta propiedad tal que:

1. **Base de inducción (TAt):** Todos los términos atómicos cumplen la propiedad $\mathbb{P}$.

   Paso de inducción:

2. (TF): Si f es un símbolo de función de aridad $n > 0$ y los términos $t_1, t_2, \ldots, t_n$ cumplen la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $f(t_1, t_2, \ldots, t_n)$ cumple la propiedad $\mathbb{P}$.

El principio de inducción estructural para términos

Como en el caso de la lógica de proposiciones, para poder estudiar las **propiedades** de las expresiones bien construidas de la lógica de primer orden una de las técnicas más adecuadas es el **principio de inducción estructural**.

Principio de inducción estructural para términos:

Sea $\mathbb{P}$ una cierta propiedad tal que:

1. **Base de inducción (TAt):** Todos los términos atómicos cumplen la propiedad $\mathbb{P}$.

   Paso de inducción:

2. (TF): Si f es un símbolo de función de aridad $n > 0$ y los términos $t_1, t_2, \ldots, t_n$ cumplen la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $f(t_1, t_2, \ldots, t_n)$ cumple la propiedad $\mathbb{P}$.

   El **principio de inducción estructural para términos** afirma que si se verifican las condiciones anteriores, entonces se puede concluir que **todo término** cumple la propiedad $\mathbb{P}$.

El principio de inducción estructural para fórmulas

Principio de inducción estructural para fórmulas:
Sea $\mathbb{P}$ una cierta propiedad tal que:

1. **Base de inducción (FAt):** Todas las fórmulas atómicas cumplen la propiedad $\mathbb{P}$.

El principio de inducción estructural para fórmulas

Principio de inducción estructural para fórmulas:
Sea $\mathbb{P}$ una cierta propiedad tal que:

1. **Base de inducción (FAt):** Todas las fórmulas atómicas cumplen la propiedad $\mathbb{P}$.

   Pasos de inducción:
2. (F$\neg$): Si φ es una fórmula que cumple la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $\neg\varphi$ cumple la propiedad $\mathbb{P}$,

El principio de inducción estructural para fórmulas

Principio de inducción estructural para fórmulas:
Sea $\mathbb{P}$ una cierta propiedad tal que:

1. **Base de inducción (FAt):** Todas las fórmulas atómicas cumplen la propiedad $\mathbb{P}$.

   Pasos de inducción:

2. (F$\neg$): Si φ es una fórmula que cumple la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $\neg\varphi$ cumple la propiedad $\mathbb{P}$,
3. (F$\circ$): Si φ y ψ son dos fórmulas que cumplen la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $(\varphi \circ \psi)$ cumple la propiedad $\mathbb{P}$,

El principio de inducción estructural para fórmulas

Principio de inducción estructural para fórmulas:
Sea $\mathbb{P}$ una cierta propiedad tal que:

1. **Base de inducción (FAt):** Todas las fórmulas atómicas cumplen la propiedad $\mathbb{P}$.

   Pasos de inducción:

2. (F$\neg$): Si φ es una fórmula que cumple la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $\neg\varphi$ cumple la propiedad $\mathbb{P}$,
3. (F$\circ$): Si φ y ψ son dos fórmulas que cumplen la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $(\varphi \circ \psi)$ cumple la propiedad $\mathbb{P}$,
4. (F$\forall\exists$): Sean φ es una fórmula y x un símbolo de variable. Si φ cumple la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $\forall x\varphi$ y $\exists x\varphi$ cumplen la propiedad $\mathbb{P}$.

El principio de inducción estructural para fórmulas

Principio de inducción estructural para fórmulas:
Sea $\mathbb{P}$ una cierta propiedad tal que:

1. **Base de inducción (FAt):** Todas las fórmulas atómicas cumplen la propiedad $\mathbb{P}$.

   Pasos de inducción:

2. (F$\neg$): Si φ es una fórmula que cumple la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $\neg\varphi$ cumple la propiedad $\mathbb{P}$,
3. (F$\circ$): Si φ y ψ son dos fórmulas que cumplen la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $(\varphi \circ \psi)$ cumple la propiedad $\mathbb{P}$,
4. (F$\forall\exists$): Sean φ es una fórmula y x un símbolo de variable. Si φ cumple la propiedad $\mathbb{P}$ (hipótesis de inducción), entonces $\forall x\varphi$ y $\exists x\varphi$ cumplen la propiedad $\mathbb{P}$.

El **principio de inducción estructural para fórmulas** afirma que si se verifican las condiciones anteriores, entonces se puede concluir que **toda fórmula** cumple la propiedad $\mathbb{P}$.

Contenido

Fórmulas en forma abreviada

Como ya hicimos en el caso de la lógica proposicional, vamos a ilustrar las maneras más usadas para representar las fórmulas de la lógica de primer orden.

Fórmulas en forma abreviada

Como ya hicimos en el caso de la lógica proposicional, vamos a ilustrar las maneras más usadas para representar las fórmulas de la lógica de primer orden.

Fórmulas en forma abreviada

Para reducir una fórmula a su **forma abreviada** podemos seguir los siguientes pasos:

Fórmulas en forma abreviada

Como ya hicimos en el caso de la lógica proposicional, vamos a ilustrar las maneras más usadas para representar las fórmulas de la lógica de primer orden.

Fórmulas en forma abreviada

Para reducir una fórmula a su **forma abreviada** podemos seguir los siguientes pasos:

a) Se puede **omitir el par de paréntesis externo**.
Así, por ejemplo, la fórmula

$$((\exists x P(x) \wedge Q(x)) \wedge (\forall y P(y)))$$

se puede escribir como:

$$(\exists x P(x) \wedge Q(x)) \wedge (\forall y P(y)).$$

Fórmulas en forma abreviada

Como ya hicimos en el caso de la lógica proposicional, vamos a ilustrar las maneras más usadas para representar las fórmulas de la lógica de primer orden.

Fórmulas en forma abreviada

Para reducir una fórmula a su **forma abreviada** podemos seguir los siguientes pasos:

a) Se puede **omitir el par de paréntesis externo**.

Así, por ejemplo, la fórmula

$$((\exists x P(x) \wedge Q(x)) \wedge (\forall y P(y)))$$

se puede escribir como:

$$(\exists x P(x) \wedge Q(x)) \wedge (\forall y P(y)).$$

Fórmulas en forma abreviada

b) Se introducen las mismas **reglas de precedencia** y el mismo **convenio de asociatividad** entre conectivos que definimos para fórmulas proposicionales.
Así, por ejemplo, la fórmula anterior se escribiría (sin cambios) como:

$$(\exists x P(x) \wedge Q(x)) \wedge (\forall y P(y)).$$

Fórmulas en forma abreviada

b) Se introducen las mismas **reglas de precedencia** y el mismo **convenio de asociatividad** entre conectivos que definimos para fórmulas proposicionales.
Así, por ejemplo, la fórmula anterior se escribiría (sin cambios) como:

$$(\exists x P(x) \wedge Q(x)) \wedge (\forall y P(y)).$$

c) Se admite la **precedencia de los cuantificadores**: los cuantificadores tienen prioridad sobre los conectivos. Se obtienen los siguientes niveles de precedencia:

Nivel 1: $\forall$ **y** $\exists$,
Nivel 2: $\neg$,
Nivel 3: $\vee$ **y** $\wedge$,
Nivel 4: $\to$ **y** $\leftrightarrow$.

Fórmulas en forma abreviada

b) Se introducen las mismas **reglas de precedencia** y el mismo **convenio de asociatividad** entre conectivos que definimos para fórmulas proposicionales.
Así, por ejemplo, la fórmula anterior se escribiría (sin cambios) como:

$$(\exists x P(x) \wedge Q(x)) \wedge (\forall y P(y)).$$

c) Se admite la **precedencia de los cuantificadores**: los cuantificadores tienen prioridad sobre los conectivos. Se obtienen los siguientes niveles de precedencia:

Nivel 1: $\forall$ **y** $\exists$,
Nivel 2: $\neg$,
Nivel 3: $\vee$ **y** $\wedge$,
Nivel 4: $\rightarrow$ **y** $\leftrightarrow$.

Finalmente, la forma abreviada de nuestra fórmula es:

$$(\exists x P(x) \wedge Q(x)) \wedge \forall y P(y).$$

Principio de unicidad de estructura

Como para la lógica proposicional, en la lógica de primer orden valen principios de unicidad de estructura para términos y fórmulas. Cada expresión bien construida admite un análisis sintáctico único, es decir, existe una única manera de derivarla usando las reglas de formación dadas.

Principio de unicidad de estructura

Como para la lógica proposicional, en la lógica de primer orden valen principios de unicidad de estructura para términos y fórmulas. Cada expresión bien construida admite un análisis sintáctico único, es decir, existe una única manera de derivarla usando las reglas de formación dadas.
Estos principios nos permitirán representar expresiones bien construidas por medio de árboles sintácticos (o estructurales).

Principio de unicidad de estructura para términos

Todo término pertenece a una y sólo una de las siguientes categorías:

Principio de unicidad de estructura para términos

Todo término pertenece a una y sólo una de las siguientes categorías:

1. (TAt): es atómico,

Principio de unicidad de estructura para términos

Todo término pertenece a una y sólo una de las siguientes categorías:

1. (TAt): es atómico,
2. (TF): se escribe como $f(t_1, t_2, \ldots, t_n)$, donde f es un símbolo de función de aridad $n > 0$ y $t_1, t_2, \ldots, t_n$ son términos.
   f y $t_1, t_2, \ldots, t_n$ están unívocamente determinados.

Principio de unicidad de estructura para fórmulas

Toda fórmula φ pertenece a una y sólo una de las siguientes categorías:

Principio de unicidad de estructura para fórmulas

Toda fórmula φ pertenece a una y sólo una de las siguientes categorías:

1. $(\text{FAt})_1$: φ es una proposición atómica o uno de los conectivos lógicos constantes $\top$ y $\perp$,

Principio de unicidad de estructura para fórmulas

Toda fórmula φ pertenece a una y sólo una de las siguientes categorías:

1. $(\text{FAt})_1$: φ es una proposición atómica o uno de los conectivos lógicos constantes $\top$ y $\bot$,
2. $(\text{FAt})_2$: φ es una igualdad entre términos, $s = t$, donde s y t están unívocamente determinados,

Principio de unicidad de estructura para fórmulas

Toda fórmula φ pertenece a una y sólo una de las siguientes categorías:

1. $(\mathrm{FAt})_1$: φ es una proposición atómica o uno de los conectivos lógicos constantes $\top$ y $\bot$,
2. $(\mathrm{FAt})_2$: φ es una igualdad entre términos, $s = t$, donde s y t están unívocamente determinados,
3. $(\mathrm{FAt})_3$: φ es $P(t_1, t_2, \ldots, t_n)$, donde P es un símbolo de predicado con aridad $n > 0$ y $t_1, t_2, \ldots, t_n$ son términos. P y $t_1, t_2, \ldots, t_n$ están unívocamente determinados,

Principio de unicidad de estructura para fórmulas

Toda fórmula φ pertenece a una y sólo una de las siguientes categorías:

1. (FAt)$_1$: φ es una proposición atómica o uno de los conectivos lógicos constantes $\top$ y $\bot$,
2. (FAt)$_2$: φ es una igualdad entre términos, $s = t$, donde s y t están unívocamente determinados,
3. (FAt)$_3$: φ es $P(t_1, t_2, \ldots, t_n)$, donde P es un símbolo de predicado con aridad $n > 0$ y $t_1, t_2, \ldots, t_n$ son términos. P y $t_1, t_2, \ldots, t_n$ están unívocamente determinados,
4. $(F\neg)$: φ es $\neg(\varphi_1)$ para cierta fórmula φ_1, unívocamente determinada,

Principio de unicidad de estructura para fórmulas

Toda fórmula φ pertenece a una y sólo una de las siguientes categorías:

1. $(\mathrm{FAt})_1$: φ es una proposición atómica o uno de los conectivos lógicos constantes $\top$ y $\bot$,
2. $(\mathrm{FAt})_2$: φ es una igualdad entre términos, $s = t$, donde s y t están unívocamente determinados,
3. $(\mathrm{FAt})_3$: φ es $P(t_1, t_2, \ldots, t_n)$, donde P es un símbolo de predicado con aridad $n > 0$ y $t_1, t_2, \ldots, t_n$ son términos. P y $t_1, t_2, \ldots, t_n$ están unívocamente determinados,
4. $(F\neg)$: φ es $\neg(\varphi_1)$ para cierta fórmula φ_1, unívocamente determinada,
5. $(F\circ)$: φ es $(\varphi_1 \circ \varphi_2)$ para cierto conectivo $\circ$ y ciertas fórmulas φ_1 y φ_2. φ_1 y φ_2 están unívocamente determinadas,

Principio de unicidad de estructura para fórmulas

Toda fórmula φ pertenece a una y sólo una de las siguientes categorías:

1. (FAt)$_1$: φ es una proposición atómica o uno de los conectivos lógicos constantes $\top$ y $\bot$,
2. (FAt)$_2$: φ es una igualdad entre términos, $s = t$, donde s y t están unívocamente determinados,
3. (FAt)$_3$: φ es $P(t_1, t_2, \ldots, t_n)$, donde P es un símbolo de predicado con aridad $n > 0$ y $t_1, t_2, \ldots, t_n$ son términos.
   P y $t_1, t_2, \ldots, t_n$ están unívocamente determinados,
4. $(F\neg)$: φ es $\neg(\varphi_1)$ para cierta fórmula φ_1, unívocamente determinada,
5. $(F\circ)$: φ es $(\varphi_1 \circ \varphi_2)$ para cierto conectivo $\circ$ y ciertas fórmulas φ_1 y φ_2 .
   φ_1 y φ_2 están unívocamente determinadas,
6. (F∀∃): φ es $\forall x \varphi_1$ o $\exists x \varphi_1$ para cierta fórmula φ_1 y cierto símbolo de variable x.
   φ_1 y x están unívocamente determinados.

Términos en forma de árbol estructural

Una primera consecuencia de los principios de unicidad de estructura es la posibilidad de representar términos y fórmulas por medio de árboles.

Términos en forma de árbol estructural

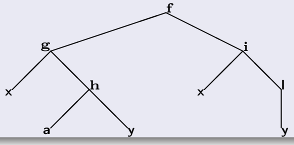
Una primera consecuencia de los principios de unicidad de estructura es la posibilidad de representar términos y fórmulas por medio de árboles. En los siguientes dos ejemplos veremos como construir el árbol sintáctico de un término y de una fórmula.

Términos en forma de árbol estructural

Una primera consecuencia de los principios de unicidad de estructura es la posibilidad de representar términos y fórmulas por medio de árboles. En los siguientes dos ejemplos veremos como construir el árbol sintáctico de un término y de una fórmula.

Ejemplos

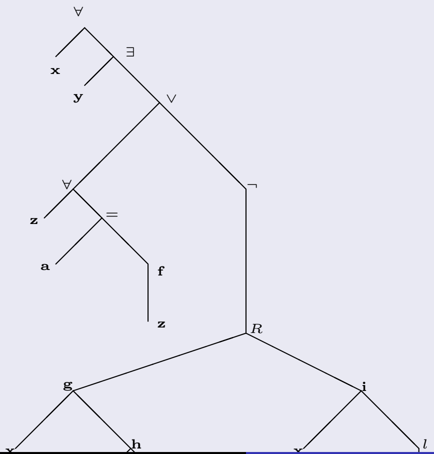
1) Consideremos el término compuesto $f(g(x, h(a, y)), i(x, l(y)))$. *Para representar su estructura sintáctica podemos dibujar el árbol con raíz de la figura:*



Términos en forma de árbol estructural

Ejemplos

2) Sea ahora $\varphi : \forall x \exists y(\forall z(a = f(z)) \vee (\neg R(g(x, h(a, y)), i(x, l(y)))))$.



El principio de recursión estructural para términos

Los principios de recursión estructural sirven para **definir funciones** sobre el conjunto de los términos y el conjuntos de las fórmulas.

El principio de recursión estructural para términos

Los principios de recursión estructural sirven para **definir funciones** sobre el conjunto de los términos y el conjuntos de las fórmulas.

El principio de recursión estructural para términos

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{T} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de los términos y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

El principio de recursión estructural para términos

Los principios de recursión estructural sirven para **definir funciones** sobre el conjunto de los términos y el conjuntos de las fórmulas.

El principio de recursión estructural para términos

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{T} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de los términos y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

1. **Base (TAt):** Si t es un término atómico, $f(t)$ se define explícitamente, es un elemento de $\mathbf{A}$,

El principio de recursión estructural para términos

Los principios de recursión estructural sirven para **definir funciones** sobre el conjunto de los términos y el conjuntos de las fórmulas.

El principio de recursión estructural para términos

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{T} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de los términos y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

1. **Base (TAt):** Si t es un término atómico, $f(t)$ se define explícitamente, es un elemento de $\mathbf{A}$,
2. **Paso recursivo (TF):** si t es el término compuesto $g(t_1, t_2, \ldots, t_n)$, se define $f(t)$ en función de $f(t_1), f(t_2), \ldots, f(t_n)$.

El principio de recursión estructural para términos

Los principios de recursión estructural sirven para **definir funciones** sobre el conjunto de los términos y el conjuntos de las fórmulas.

El principio de recursión estructural para términos

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{T} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de los términos y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

1. **Base (TAt):** Si t es un término atómico, $f(t)$ se define explícitamente, es un elemento de $\mathbf{A}$,
2. **Paso recursivo (TF):** si t es el término compuesto $g(t_1, t_2, \ldots, t_n)$, se define $f(t)$ en función de $f(t_1), f(t_2), \ldots, f(t_n)$.

Estas definiciones determinan la función f sobre todo $\mathbf{T}$.

El principio de recursión estructural para términos

Ejemplos

1) Sea $\mathbf{Var}$ *el conjunto de todas las variables. Queremos definir recursivamente el conjunto* $Var(t)$ *de las variables libres de un cualquier término* t.

Por el principio de recursión estructural para términos, podemos definir una función $Var : \mathbf{T} \longrightarrow P(\mathbf{Var})$, *(*$P(\mathbf{Var})$ *es el conjunto de las partes de* $\mathbf{Var}$*) que asocia a todo término* t *el conjunto de sus variables libres,* $Var(t)$:

El principio de recursión estructural para términos

Ejemplos

1) Sea $\mathbf{Var}$ *el conjunto de todas las variables. Queremos definir recursivamente el conjunto* $Var(t)$ *de las variables libres de un cualquier término* t.
Por el principio de recursión estructural para términos, podemos definir una función $Var : \mathbf{T} \longrightarrow P(\mathbf{Var})$, *(*$P(\mathbf{Var})$ *es el conjunto de las partes de* $\mathbf{Var}$*) que asocia a todo término* t *el conjunto de sus variables libres,* $Var(t)$:

1. **Base (TAt):** $Var(x) = \{x\}$, *y* $Var(a) = \emptyset$, *donde* $\emptyset$ *es el conjunto vacío.*

El principio de recursión estructural para términos

Ejemplos

1) Sea $\mathbf{Var}$ *el conjunto de todas las variables. Queremos definir recursivamente el conjunto* $Var(t)$ *de las variables libres de un cualquier término* t.
Por el principio de recursión estructural para términos, podemos definir una función $Var : \mathbf{T} \longrightarrow P(\mathbf{Var})$, *(*$P(\mathbf{Var})$ *es el conjunto de las partes de* $\mathbf{Var}$*) que asocia a todo término* t *el conjunto de sus variables libres,* $Var(t)$:

1. **Base (TAt):** $Var(x) = \{x\}$, *y* $Var(a) = \emptyset$, *donde* $\emptyset$ *es el conjunto vacío.*
2. **Paso recursivo (TF):** *si* t *es el término compuesto* $g(t_1, t_2, \ldots, t_n)$, *se define*

$$Var(t) = Var(t_1) \cup Var(t_2) \cup \cdots \cup Var(t_n).$$

El principio de recursión estructural para términos

Ejemplos

2) Se denomina **profundidad** *de un término t a la longitud de la rama más larga del árbol estructural de t. Por tanto, la profundidad de t es la altura de su árbol sintáctico.*
Por el principio de recursión estructural, la definición recursiva de la función profundidad

$$pf : \mathbf{T} \longrightarrow \mathbb{N} \cup \{\mathbf{0}\}$$

(que asocia a cada término $t \in \mathbf{T}$ su profundidad) es:

El principio de recursión estructural para términos

Ejemplos

2) Se denomina **profundidad** *de un término* t *a la longitud de la rama más larga del árbol estructural de* t*. Por tanto, la profundidad de* t *es la altura de su árbol sintáctico.*
Por el principio de recursión estructural, la definición recursiva de la función profundidad

$$pf : \mathbf{T} \longrightarrow \mathbb{N} \cup \{\mathbf{0}\}$$

(que asocia a cada término $t \in \mathbf{T}$ *su profundidad) es:*

1. **Base (TAt):** *Si* t *es un término atómico,* $pf(t) = 0$,
2. **Paso recursivo (TF):** *si* t *es el término compuesto* $g(t_1, t_2, \ldots, t_n)$, *se define*

$$pf(t) = max\{pf(t_1), pf(t_2), \ldots, pf(t_n)\} + 1.$$

El principio de recursión estructural para fórmulas

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{F} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de las fórmulas y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

El principio de recursión estructural para fórmulas

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{F} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de las fórmulas y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

1. **Base (FAt):** Si φ es una fórmula atómica, $f(\varphi)$ se define explícitamente, es un elemento de $\mathbf{A}$.

El principio de recursión estructural para fórmulas

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{F} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de las fórmulas y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

1. **Base (FAt):** Si φ es una fórmula atómica, $f(\varphi)$ se define explícitamente, es un elemento de $\mathbf{A}$.

   Pasos recursivos:

2. (F¬): $f(\neg(\varphi_1))$ se define en función de $f(\varphi_1)$,

El principio de recursión estructural para fórmulas

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{F} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de las fórmulas y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

1. **Base (FAt):** Si φ es una fórmula atómica, $f(\varphi)$ se define explícitamente, es un elemento de $\mathbf{A}$.

   Pasos recursivos:

2. (F¬): $f(\neg(\varphi_1))$ se define en función de $f(\varphi_1)$,
3. (F∘): $f(\varphi_1 \circ \varphi_2)$ se define en función de $f(\varphi_1)$ y $f(\varphi_2)$,

El principio de recursión estructural para fórmulas

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{F} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de las fórmulas y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

1. **Base (FAt):** Si φ es una fórmula atómica, $f(\varphi)$ se define explícitamente, es un elemento de $\mathbf{A}$.

   Pasos recursivos:

2. (F¬): $f(\neg(\varphi_1))$ se define en función de $f(\varphi_1)$,
3. (F∘): $f(\varphi_1 \circ \varphi_2)$ se define en función de $f(\varphi_1)$ y $f(\varphi_2)$,
4. (F∀∃): $f(\forall x \varphi_1)$ y $f(\exists x \varphi_1)$ se definen en función de $f(\varphi_1)$.

El principio de recursión estructural para fórmulas

Sea A un conjunto no vacío. Para definir una función

$$f : \mathbf{F} \longrightarrow \mathbf{A}$$

cuyo dominio sea el conjunto de las fórmulas y cuyo codominio sea un conjunto dado $\mathbf{A}$, se puede emplear la siguiente definición recursiva:

1. **Base (FAt):** Si φ es una fórmula atómica, $f(\varphi)$ se define explícitamente, es un elemento de $\mathbf{A}$.

   Pasos recursivos:
2. (F¬): $f(\neg(\varphi_1))$ se define en función de $f(\varphi_1)$,
3. (F∘): $f(\varphi_1 \circ \varphi_2)$ se define en función de $f(\varphi_1)$ y $f(\varphi_2)$,
4. (F∀∃): $f(\forall x \varphi_1)$ y $f(\exists x \varphi_1)$ se definen en función de $f(\varphi_1)$.

Estas definiciones determinan la función f sobre todo $\mathbf{F}$.

El principio de recursión estructural para fórmulas

Ejemplo

Sea $\mathbf{Var}$ *el conjunto de todas las variables. Queremos definir recursivamente el conjunto* $Lib(\varphi)$ *de las variables libres de una cualquier fórmula* φ.

El principio de recursión estructural para fórmulas

Ejemplo

Sea $\mathbf{Var}$ *el conjunto de todas las variables. Queremos definir recursivamente el conjunto* $Lib(\varphi)$ *de las variables libres de una cualquier fórmula* φ.
Por el principio de recursión estructural para fórmulas, podemos definir una función $Lib : \mathbf{F} \longrightarrow P(\mathbf{Var})$, *que asocia a toda fórmula* φ *el conjunto de sus variables libres,* $Lib(\varphi)$:

El principio de recursión estructural para fórmulas

Ejemplo

Sea $\mathbf{Var}$ *el conjunto de todas las variables. Queremos definir recursivamente el conjunto* $Lib(\varphi)$ *de las variables libres de una cualquier fórmula* φ.
Por el principio de recursión estructural para fórmulas, podemos definir una función $Lib : \mathbf{F} \longrightarrow P(\mathbf{Var})$, *que asocia a toda fórmula* φ *el conjunto de sus variables libres,* $Lib(\varphi)$:

1. **Base (FAt):** *Si* φ *es una fórmula atómica,* $Lib(\varphi)$ *es el conjunto* $Var(\varphi)$ *de todas las variables de* φ. *En particular,*
   $Lib(\top) = Lib(\bot) = Lib(p) = \emptyset$, *donde* $\emptyset$ *es el conjunto vacío,*
   $Lib(s = t) = Var(s) \cup Var(t)$,
   $Lib(P(t_1, t_2, \ldots, t_n)) = Var(t_1) \cup Var(t_2) \cup \cdots \cup Var(t_n)$.

El principio de recursión estructural para fórmulas

Ejemplo

Sea $\mathbf{Var}$ *el conjunto de todas las variables. Queremos definir recursivamente el conjunto* $Lib(\varphi)$ *de las variables libres de una cualquier fórmula* φ.
Por el principio de recursión estructural para fórmulas, podemos definir una función $Lib : \mathbf{F} \longrightarrow P(\mathbf{Var})$, *que asocia a toda fórmula* φ *el conjunto de sus variables libres,* $Lib(\varphi)$:

1. **Base (FAt):** *Si* φ *es una fórmula atómica,* $Lib(\varphi)$ *es el conjunto* $Var(\varphi)$ *de todas las variables de* φ. *En particular,*
   $Lib(\top) = Lib(\bot) = Lib(p) = \emptyset$, *donde* $\emptyset$ *es el conjunto vacío,*
   $Lib(s = t) = Var(s) \cup Var(t)$,
   $Lib(P(t_1, t_2, \ldots, t_n)) = Var(t_1) \cup Var(t_2) \cup \cdots \cup Var(t_n)$.

El principio de recursión estructural para fórmulas

Ejemplo

Pasos recursivos:

1. *(F¬)*: $Lib(\neg(\varphi_1)) = Lib(\varphi_1)$,

El principio de recursión estructural para fórmulas

Ejemplo

Pasos recursivos:

1. *(F¬):* $Lib(\neg(\varphi_1)) = Lib(\varphi_1)$,
2. *(F∘):* $Lib(\varphi_1 \circ \varphi_2) = Lib(\varphi_1) \cup Lib(\varphi_2)$,

El principio de recursión estructural para fórmulas

Ejemplo

Pasos recursivos:

1. *(F¬):* $Lib(\neg(\varphi_1)) = Lib(\varphi_1)$,
2. *(F∘):* $Lib(\varphi_1 \circ \varphi_2) = Lib(\varphi_1) \cup Lib(\varphi_2)$,
3. *(F∀∃):* $Lib(\forall x \varphi_1) = Lib(\exists x \varphi_1) = Lib(\varphi_1) \setminus \{x\}$.

El principio de recursión estructural para fórmulas

Ejemplo

Pasos recursivos:

1. *(F¬):* $Lib(\neg(\varphi_1)) = Lib(\varphi_1)$,
2. *(F∘):* $Lib(\varphi_1 \circ \varphi_2) = Lib(\varphi_1) \cup Lib(\varphi_2)$,
3. *(F∀∃):* $Lib(\forall x \varphi_1) = Lib(\exists x \varphi_1) = Lib(\varphi_1) \setminus \{x\}$.

Estas definiciones determinan la función Lib sobre todo $\mathbf{F}$.

Contenido

Formalización del lenguaje natural

Como ya vimos en lógica proposicional, formalizar una frase del lenguaje natural consiste en encontrar una expresión que la represente fielmente en el lenguaje formal.

Formalización del lenguaje natural

Como ya vimos en lógica proposicional, formalizar una frase del lenguaje natural consiste en encontrar una expresión que la represente fielmente en el lenguaje formal.
No hay procedimientos generales para la formalización, pero se pueden determinar algunas estrategias, como las que vamos a indicar a continuación.

Formalización del lenguaje natural

Como ya vimos en lógica proposicional, formalizar una frase del lenguaje natural consiste en encontrar una expresión que la represente fielmente en el lenguaje formal.
No hay procedimientos generales para la formalización, pero se pueden determinar algunas estrategias, como las que vamos a indicar a continuación.

Formalización del lenguaje natural

- Si la frase que se quiere formalizar no tiene una estructura sintáctica fácilmente reconocible, se puede intentar **reescribirla** en el lenguaje natural hasta llegar a una frase con una estructura más sencilla y que mantenga el mismo significado.

Formalización del lenguaje natural

- Si la frase que se quiere formalizar no tiene una estructura sintáctica fácilmente reconocible, se puede intentar **reescribirla** en el lenguaje natural hasta llegar a una frase con una estructura más sencilla y que mantenga el mismo significado.
- Tenemos que **definir claramente el dominio o los dominios** a los cuáles pertenecen los objetos que vamos a usar.

Formalización del lenguaje natural

- Si la frase que se quiere formalizar no tiene una estructura sintáctica fácilmente reconocible, se puede intentar **reescribirla** en el lenguaje natural hasta llegar a una frase con una estructura más sencilla y que mantenga el mismo significado.
- Tenemos que **definir claramente el dominio o los dominios** a los cuáles pertenecen los objetos que vamos a usar.
- En una frase necesitamos determinar:
  - Las **constantes**, que son objetos concretos de uno o más dominios.

Formalización del lenguaje natural

- Si la frase que se quiere formalizar no tiene una estructura sintáctica fácilmente reconocible, se puede intentar **reescribirla** en el lenguaje natural hasta llegar a una frase con una estructura más sencilla y que mantenga el mismo significado.
- Tenemos que **definir claramente el dominio o los dominios** a los cuáles pertenecen los objetos que vamos a usar.
- En una frase necesitamos determinar:
  - Las **constantes**, que son objetos concretos de uno o más dominios.
  - Las **variables**, que son objetos genéricos de uno o más dominios.

Formalización del lenguaje natural

- Si la frase que se quiere formalizar no tiene una estructura sintáctica fácilmente reconocible, se puede intentar **reescribirla** en el lenguaje natural hasta llegar a una frase con una estructura más sencilla y que mantenga el mismo significado.
- Tenemos que **definir claramente el dominio o los dominios** a los cuáles pertenecen los objetos que vamos a usar.
- En una frase necesitamos determinar:
  - Las **constantes**, que son objetos concretos de uno o más dominios.
  - Las **variables**, que son objetos genéricos de uno o más dominios.
  - Las **funciones de aridad** $n > 0$, que representan como un cierto objeto queda determinado por otros (u otro).

Formalización del lenguaje natural

- Si la frase que se quiere formalizar no tiene una estructura sintáctica fácilmente reconocible, se puede intentar **reescribirla** en el lenguaje natural hasta llegar a una frase con una estructura más sencilla y que mantenga el mismo significado.
- Tenemos que **definir claramente el dominio o los dominios** a los cuáles pertenecen los objetos que vamos a usar.
- En una frase necesitamos determinar:
  - Las **constantes**, que son objetos concretos de uno o más dominios.
  - Las **variables**, que son objetos genéricos de uno o más dominios.
  - Las **funciones de aridad** $n > 0$, que representan como un cierto objeto queda determinado por otros (u otro).
  - Los **predicados monádicos** que representan propiedades de un objeto.

Formalización del lenguaje natural

- Si la frase que se quiere formalizar no tiene una estructura sintáctica fácilmente reconocible, se puede intentar **reescribirla** en el lenguaje natural hasta llegar a una frase con una estructura más sencilla y que mantenga el mismo significado.
- Tenemos que **definir claramente el dominio o los dominios** a los cuáles pertenecen los objetos que vamos a usar.
- En una frase necesitamos determinar:
    - Las **constantes**, que son objetos concretos de uno o más dominios.
    - Las **variables**, que son objetos genéricos de uno o más dominios.
    - Las **funciones de aridad** $n > 0$, que representan como un cierto objeto queda determinado por otros (u otro).
    - Los **predicados monádicos** que representan propiedades de un objeto.
    - Los **predicados de aridad** $n > 0$ que representan relaciones entre objetos.

Formalización del lenguaje natural

- Si la frase que se quiere formalizar no tiene una estructura sintáctica fácilmente reconocible, se puede intentar **reescribirla** en el lenguaje natural hasta llegar a una frase con una estructura más sencilla y que mantenga el mismo significado.
- Tenemos que **definir claramente el dominio o los dominios** a los cuáles pertenecen los objetos que vamos a usar.
- En una frase necesitamos determinar:
  - Las **constantes**, que son objetos concretos de uno o más dominios.
  - Las **variables**, que son objetos genéricos de uno o más dominios.
  - Las **funciones de aridad** $n > 0$, que representan como un cierto objeto queda determinado por otros (u otro).
  - Los **predicados monádicos** que representan propiedades de un objeto.
  - Los **predicados de aridad** $n > 0$ que representan relaciones entre objetos.
- Identificadas las conectivas lingüísticas y los cuantificadores (universales o existenciales), necesitamos sustituirlas por los **conectivos** y los **cuantificadores** de la lógica de primer orden.

Formalización del lenguaje natural

- Para formalizar un **razonamiento**, necesitamos formalizar el conjunto de sus premisas y de su conclusión.

Formalización del lenguaje natural

- Para formalizar un **razonamiento**, necesitamos formalizar el conjunto de sus premisas y de su conclusión.

A la hora de formalizar frases o razonamientos hay algunas observaciones importantes que es oportuno tener en cuenta.

Formalización del lenguaje natural

- Para formalizar un **razonamiento**, necesitamos formalizar el conjunto de sus premisas y de su conclusión.

A la hora de formalizar frases o razonamientos hay algunas observaciones importantes que es oportuno tener en cuenta.

- Ya que la formalización de una frase depende del dominio o de los dominios elegidos, se pueden obtener formalizaciones distintas de un mismo enunciado.

Formalización del lenguaje natural

- Para formalizar un **razonamiento**, necesitamos formalizar el conjunto de sus premisas y de su conclusión.

A la hora de formalizar frases o razonamientos hay algunas observaciones importantes que es oportuno tener en cuenta.

- Ya que la formalización de una frase depende del dominio o de los dominios elegidos, se pueden obtener formalizaciones distintas de un mismo enunciado.

Ejemplo

Para formalizar la frase "Todos los niños juegan con la pelota," podemos definir los predicados $J(x)$: x juega con la pelota y $J(x, y)$: x juega con y.

Formalización del lenguaje natural

- Para formalizar un **razonamiento**, necesitamos formalizar el conjunto de sus premisas y de su conclusión.

A la hora de formalizar frases o razonamientos hay algunas observaciones importantes que es oportuno tener en cuenta.

- Ya que la formalización de una frase depende del dominio o de los dominios elegidos, se pueden obtener formalizaciones distintas de un mismo enunciado.

Ejemplo

Para formalizar la frase "Todos los niños juegan con la pelota," podemos definir los predicados $J(x)$: x *juega con la pelota y*
$J(x, y)$: x *juega con* y.
a) Sea D_1 *el conjunto de los niños. Entonces se obtiene* $\forall x J(x)$.

Formalización del lenguaje natural

- Para formalizar un **razonamiento**, necesitamos formalizar el conjunto de sus premisas y de su conclusión.

A la hora de formalizar frases o razonamientos hay algunas observaciones importantes que es oportuno tener en cuenta.

- Ya que la formalización de una frase depende del dominio o de los dominios elegidos, se pueden obtener formalizaciones distintas de un mismo enunciado.

Ejemplo

Para formalizar la frase "Todos los niños juegan con la pelota," podemos definir los predicados $J(x)$: x juega con la pelota y
$J(x,y)$: *x juega con y.*
a) Sea D_1 el conjunto de los niños. Entonces se obtiene $\forall x J(x)$.
b) Sean D_2 el conjunto de las personas y sea "$N(x)$: x es un niño." En este caso se obtiene $\forall x(N(x) \rightarrow J(x))$.

Formalización del lenguaje natural

Ejemplo

c) Sean D_1 el conjunto de los niños y D_2 el conjunto de los juegos. Entonces "$p =$ la pelota" es una constante en D_2 y obtenemos la formalización $\forall x J(x, p)$.

Formalización del lenguaje natural

Ejemplo

c) Sean D_1 *el conjunto de los niños y* D_2 *el conjunto de los juegos. Entonces "*$p =$ *la pelota" es una constante en* D_2 *y obtenemos la formalización* $\forall x J(x,p)$.

d) Sean D_1 *el conjunto de las personas y* D_2 *el conjunto de los juegos. Entonces "*$p =$ *la pelota" es una constante en* D_2 *y, usando el predicado "*$N(x) : x$ *es un niño," obtenemos la formalización* $\forall x(N(x) \rightarrow J(x,p))$.

Formalización del lenguaje natural

- Toda función se puede representar mediante un predicado con un argumento más que la función.

Formalización del lenguaje natural

- Toda función se puede representar mediante un predicado con un argumento más que la función.

Ejemplo

Consideremos la frase: "Todo padre quiere mucho a sus hijos."

Formalización del lenguaje natural

- Toda función se puede representar mediante un predicado con un argumento más que la función.

Ejemplo

Consideremos la frase: "Todo padre quiere mucho a sus hijos."
a) Formalización con predicados.
Podemos definir el dominio D de las personas y los predicados $P(x, y)$: x es el padre de y, y $Q(x, y)$: x quiere mucho a y. Con estas definiciones, la formalización sería $\forall x \forall y (P(x, y) \rightarrow Q(x, y))$.

Formalización del lenguaje natural

- Toda función se puede representar mediante un predicado con un argumento más que la función.

Ejemplo

Consideremos la frase: "Todo padre quiere mucho a sus hijos."
a) Formalización con predicados.
Podemos definir el dominio D de las personas y los predicados $P(x, y)$: x es el padre de y, y $Q(x, y)$: x quiere mucho a y. Con estas definiciones, la formalización sería $\forall x \forall y (P(x, y) \rightarrow Q(x, y))$.
b) Formalización con funciones.
Podemos definir el dominio D de las personas, la función $f(x)$: el padre de x, y $Q(x, y)$: x quiere mucho a y. Con estas definiciones, la nueva formalización sería $\forall x (Q(f(x), x))$. Notar que la formalización se ha simplificado y que la función de un argumento $f(x)$ sustituye al predicado binario $P(x, y)$.

Formalización del lenguaje natural

- Toda función se puede representar mediante un predicado con un argumento más que la función.

Ejemplo

Consideremos la frase: "Todo padre quiere mucho a sus hijos."
a) Formalización con predicados.
Podemos definir el dominio D de las personas y los predicados $P(x, y)$: x es el padre de y, y $Q(x, y)$: x quiere mucho a y. Con estas definiciones, la formalización sería $\forall x \forall y(P(x, y) \rightarrow Q(x, y))$.
b) Formalización con funciones.
Podemos definir el dominio D de las personas, la función $f(x)$: el padre de x, y $Q(x, y)$: x quiere mucho a y. Con estas definiciones, la nueva formalización sería $\forall x(Q(f(x), x))$. Notar que la formalización se ha simplificado y que la función de un argumento $f(x)$ sustituye al predicado binario $P(x, y)$.
Observar también que "el hijo de x" no es una función, ya que un mismo padre puede tener más que un hijo y, por tanto, el término asociado a x (al padre) no quedaría unívocamente determinado.

Patrones en la formalización de los cuantificadores

- **Universal afirmativo**

$$\forall x(\varphi_1 \rightarrow \varphi_2),$$

$$\forall x(\neg\varphi_2 \rightarrow \neg\varphi_1).$$

Es la forma de representar frases del tipo:

Patrones en la formalización de los cuantificadores

- **Universal afirmativo**

$$\forall x(\varphi_1 \to \varphi_2),$$

$$\forall x(\neg\varphi_2 \to \neg\varphi_1).$$

Es la forma de representar frases del tipo:
Todo φ_1 es φ_2,
Sólo los φ_2 son φ_1,
Nadie es φ_1 a menos que sea φ_2,
No hay ningún φ_1 que no sea φ_2,
φ_1 es suficiente para φ_2,
φ_2 es necesario para φ_1.

Patrones en la formalización de los cuantificadores

- **Universal afirmativo**

$$\forall x(\varphi_1 \rightarrow \varphi_2),$$

$$\forall x(\neg\varphi_2 \rightarrow \neg\varphi_1).$$

  Es la forma de representar frases del tipo:
  Todo φ_1 es φ_2,
  Sólo los φ_2 son φ_1,
  Nadie es φ_1 a menos que sea φ_2,
  No hay ningún φ_1 que no sea φ_2,
  φ_1 es suficiente para φ_2,
  φ_2 es necesario para φ_1.

- **Universal negativo**

$$\forall x(\varphi_1 \rightarrow \neg\varphi_2).$$

  Es la forma de representar frases del tipo:

Patrones en la formalización de los cuantificadores

- **Universal afirmativo**

$$\forall x(\varphi_1 \to \varphi_2),$$

$$\forall x(\neg\varphi_2 \to \neg\varphi_1).$$

Es la forma de representar frases del tipo:
Todo φ_1 es φ_2,
Sólo los φ_2 son φ_1,
Nadie es φ_1 a menos que sea φ_2,
No hay ningún φ_1 que no sea φ_2,
φ_1 es suficiente para φ_2,
φ_2 es necesario para φ_1.

- **Universal negativo**

$$\forall x(\varphi_1 \to \neg\varphi_2).$$

Es la forma de representar frases del tipo:
Ningún φ_1 es φ_2,
Todos los φ_1 carecen de φ_2.

Patrones en la formalización de los cuantificadores

- **Existencial afirmativo**

$$\exists x(\varphi_1 \wedge \varphi_2).$$

Es la forma de representar frases del tipo:

Patrones en la formalización de los cuantificadores

- **Existencial afirmativo**

$$\exists x(\varphi_1 \wedge \varphi_2).$$

Es la forma de representar frases del tipo:
Algún φ_1 es φ_2,
Alguien es a la vez φ_1 y φ_2.

Patrones en la formalización de los cuantificadores

- **Existencial afirmativo**

$$\exists x(\varphi_1 \wedge \varphi_2).$$

  Es la forma de representar frases del tipo:
  Algún φ_1 es φ_2,
  Alguien es a la vez φ_1 y φ_2.

- **Existencial negativo**

$$\exists x(\varphi_1 \wedge \neg\varphi_2).$$

  Es la forma de representar frases del tipo:

Patrones en la formalización de los cuantificadores

- **Existencial afirmativo**

$$\exists x(\varphi_1 \wedge \varphi_2).$$

  Es la forma de representar frases del tipo:
  Algún φ_1 es φ_2,
  Alguien es a la vez φ_1 y φ_2.

- **Existencial negativo**

$$\exists x(\varphi_1 \wedge \neg\varphi_2).$$

  Es la forma de representar frases del tipo:
  Algún φ_1 no es φ_2,
  No todos los φ_1 son φ_2.

Patrones en la formalización de los cuantificadores

Ejemplos

1) (Universal afirmativo) "Nadie se levanta a menos que tenga que irse." La frase anterior se puede reescribir como "Para todo x, si x no tiene que irse, entonces no se levanta," o como "Para todo x, si x se levanta, entonces tiene que irse."
Sea D el dominio de las personas y sean $P(x)$: x se levanta, $Q(x)$: x tiene que irse. Con estas definiciones obtenemos la formalización:

$$\forall x\,(P(x) \rightarrow Q(x)).$$

Patrones en la formalización de los cuantificadores

Ejemplos

1) (Universal afirmativo) "Nadie se levanta a menos que tenga que irse." La frase anterior se puede reescribir como "Para todo x, si x no tiene que irse, entonces no se levanta," o como "Para todo x, si x se levanta, entonces tiene que irse."
Sea D el dominio de las personas y sean $P(x)$: x se levanta, $Q(x)$: x tiene que irse. Con estas definiciones obtenemos la formalización:

$$\forall x\,(P(x) \rightarrow Q(x)).$$

2) (Universal negativo)
"Ningún emperador es odontólogo (L. Carroll)."
Sea D el dominio de las personas y sean $P(x)$: x es emperador, $Q(x)$: x es odontólogo. Con estas definiciones obtenemos la formalización: $\forall x\,(P(x) \rightarrow \neg Q(x))$.

Patrones en la formalización de los cuantificadores

Ejemplos

3) (Existencial afirmativo)
"Algunos estudiantes de informática sólo son amigos de los aficionados a la lógica."
Esta frase se puede reescribir como: "Para algunos estudiantes de informática, una persona es un amigo sólo si es aficionado a la lógica."
Sea D el dominio de las personas y sean $P(x)$: x es estudiante de informática, $Q(x)$: x es aficionado a la lógica, $R(x, y)$: x es amigo de y. Con estas definiciones obtenemos la formalización: $\exists x\,(P(x) \wedge \forall y(R(x, y) \rightarrow Q(y)))$.

Patrones en la formalización de los cuantificadores

Ejemplos

3) (Existencial afirmativo)
"Algunos estudiantes de informática sólo son amigos de los aficionados a la lógica."
Esta frase se puede reescribir como: "Para algunos estudiantes de informática, una persona es un amigo sólo si es aficionado a la lógica."
Sea D el dominio de las personas y sean $P(x)$: x es estudiante de informática, $Q(x)$: x es aficionado a la lógica, $R(x, y)$: x es amigo de y. Con estas definiciones obtenemos la formalización: $\exists x\,(P(x) \wedge \forall y(R(x, y) \rightarrow Q(y)))$.
4) (Existencial negativo)
"Algunos gatos no saben silbar ni maullar (L. Carroll)."
Sea D el dominio de los animales y sean $P(x)$: x es un gato, $Q(x)$: x sabe silbar, $R(x)$: x sabe maullar. Con estas definiciones obtenemos la formalización: $\exists x\,(P(x) \wedge \neg Q(x) \wedge \neg R(x))$.

Negación de frases que contienen cuantificadores

Vamos a ver como se escribe **la negación** de una frase que contiene un cuantificador.

Negación de frases que contienen cuantificadores

Vamos a ver como se escribe **la negación** de una frase que contiene un cuantificador.

Universal-Existencial

Consideremos la frase "Todos los alumnos de esta clase aprobarán en febrero."
Sean D el conjunto de los alumnos de esta clase y
$P(x)$: x aprobará en febrero. La frase dada se puede escribir como:
$\forall x P(x)$.

Negación de frases que contienen cuantificadores

Vamos a ver como se escribe **la negación** de una frase que contiene un cuantificador.

Universal-Existencial

Consideremos la frase "Todos los alumnos de esta clase aprobarán en febrero."
Sean D el conjunto de los alumnos de esta clase y
$P(x)$: x aprobará en febrero. La frase dada se puede escribir como: $\forall x P(x)$.
La negación de "Todos los alumnos de esta clase aprobarán en febrero" es "No todos los alumnos de esta clase aprobarán en febrero," es decir, $\neg(\forall x P(x))$, que podemos reescribir como: "Existen alumnos de esta clase que no aprobarán en febrero."
Con los mismos dominio y predicados anteriores, su formalización es $\exists x(\neg P(x))$.

Negación de frases que contienen cuantificadores

Existencial-Universal

Consideremos ahora la frase “Algunos alumnos de esta clase suspenderán en febrero.”
Sean D el conjunto de los alumnos de esta clase y
$P(x)$: x suspenderá en febrero. La frase dada se puede escribir como:
$\exists x P(x)$.

Negación de frases que contienen cuantificadores

Existencial-Universal

Consideremos ahora la frase "Algunos alumnos de esta clase suspenderán en febrero."
Sean D el conjunto de los alumnos de esta clase y
$P(x)$: x suspenderá en febrero. La frase dada se puede escribir como: $\exists x P(x)$.
La negación de "Algunos alumnos de esta clase suspenderán en febrero" es "Ningún alumno de esta clase suspenderá en febrero," es decir, $\neg(\exists x P(x))$, que podemos reescribir como: "Todos los alumnos de esta clase no suspenderán en febrero."
Con los mismos dominio y predicados anteriores, su formalización es $\forall x(\neg P(x))$.

Negación de frases que contienen cuantificadores

Ejemplo

(Definición de límite de una sucesión) Sean $\mathbb{R}$ *el conjunto de los números reales,* $\mathbb{R}^+$ *el conjunto de los números reales positivos y* $\mathbb{N}$ *el conjunto de los números naturales. Sea* $\{a_n\}_{n\in\mathbb{N}}$ *una sucesión de números reales, es decir, una función* $a : \mathbb{N} \longrightarrow \mathbb{R}$. *Si* $\varepsilon \in \mathbb{R}^+$ *y* $n, m \in \mathbb{N}$, *se dice que el número real* L *es el límite de* $\{a_n\}_{n\in\mathbb{N}}$, $\lim_{n\to\infty} a_n = L$, *si*

$\forall\varepsilon\,\exists n\,\forall m\,((m \geq n) \to |a_m - L| < \varepsilon)$.

La formalización matemática de la condición $\lim_{n\to\infty} a_n \neq L$ *es*

$\exists\varepsilon\,\forall n\,\exists m\,((m \geq n) \wedge |a_m - L| \geq \varepsilon)$.

Formalización de un razonamiento

Ejemplo

Vamos a formalizar el siguiente razonamiento:
"Sólo las buenas personas ayudan a los pobres. Ninguna buena persona es aficionada a la fotografía. Antonio ayuda a Juan. Antonio es aficionado a la fotografía. Entonces, Juan es pobre."

Formalización de un razonamiento

Ejemplo

Vamos a formalizar el siguiente razonamiento:
"Sólo las buenas personas ayudan a los pobres. Ninguna buena persona es aficionada a la fotografía. Antonio ayuda a Juan. Antonio es aficionado a la fotografía. Entonces, Juan es pobre."
Sea D el dominio de las personas, a la constante Antonio y j la constante Juan. Definamos los siguientes predicados:
$P(x)$: x *es buena persona*, $Q(x, y)$: x *ayuda a* y, $R(x)$: x *es pobre*,
$S(x)$: x *es aficionado a la fotografía.*

Formalización de un razonamiento

Ejemplo

Vamos a formalizar el siguiente razonamiento:
"Sólo las buenas personas ayudan a los pobres. Ninguna buena persona es aficionada a la fotografía. Antonio ayuda a Juan. Antonio es aficionado a la fotografía. Entonces, Juan es pobre."
Sea D el dominio de las personas, a la constante Antonio y j la constante Juan. Definamos los siguientes predicados:
$P(x)$: x *es buena persona*, $Q(x, y)$: x *ayuda a* y, $R(x)$: x *es pobre*,
$S(x)$: x *es aficionado a la fotografía.*
Con estas definiciones el razonamiento dado se puede escribir como:

$$\forall x \forall y (Q(x, y) \wedge R(y) \rightarrow P(x)),$$

$$\forall x (P(x) \rightarrow \neg S(x)),$$

$$Q(a, j),$$

$$S(a)$$

$$\overline{}$$

$$R(j)$$

Lógicas de predicados de orden superior

Observación

(Lógicas de predicados de orden superior)
El cálculo de predicados de primer orden admite generalizaciones a cálculos de predicados de orden mayor que uno. En el cálculo de predicados de primer orden los cuantificadores pueden afectar sólo a las variables y los predicados se calculan sólo sobre términos.

Lógicas de predicados de orden superior

Observación

(Lógicas de predicados de orden superior)
El cálculo de predicados de primer orden admite generalizaciones a cálculos de predicados de orden mayor que uno. En el cálculo de predicados de primer orden los cuantificadores pueden afectar sólo a las variables y los predicados se calculan sólo sobre términos.
En el cálculo de predicados de segundo orden, los cuantificadores afectan también a predicados.

Lógicas de predicados de orden superior

Observación

(Lógicas de predicados de orden superior)
El cálculo de predicados de primer orden admite generalizaciones a cálculos de predicados de orden mayor que uno. En el cálculo de predicados de primer orden los cuantificadores pueden afectar sólo a las variables y los predicados se calculan sólo sobre términos.
En el cálculo de predicados de segundo orden, los cuantificadores afectan también a predicados.
En el cálculo de tercer orden se definen predicados de predicados (no sólo predicados de términos).

Lógicas de predicados de orden superior

Observación

(Lógicas de predicados de orden superior)
El cálculo de predicados de primer orden admite generalizaciones a cálculos de predicados de orden mayor que uno. En el cálculo de predicados de primer orden los cuantificadores pueden afectar sólo a las variables y los predicados se calculan sólo sobre términos.
En el cálculo de predicados de segundo orden, los cuantificadores afectan también a predicados.
En el cálculo de tercer orden se definen predicados de predicados (no sólo predicados de términos).
Siguiendo añadiendo niveles de "predicados de predicados," se sube el nivel del cálculo de predicados que se está definiendo.