Prolog El Lenguaje de Programación Lógica.

INTRODUCCIÓN

En este artículo se introducirá el tema del lenguaje de programación Prolog, hablare sobre su estructura, sintaxis, los diferentes tipos de cláusulas como lo son los hechos y las reglas.

HISTORIA

Creado a principios de los años 70 en la Universidad de Aix-Marseille I (Marsella, Francia) por Alain Colmerauer y Philippe Roussel.

La difusión del lenguaje se produce en los 80, pero de forma muy limitada debido a la falta de aplicaciones en dicho lenguaje.

Nació de un proyecto que no tenía como objetivo la traducción de un lenguaje de programación, sino el tratamiento algorítmico de lenguajes naturales.

Prolog se enmarca en el paradigma de los lenguajes lógicos y declarativos, lo que lo diferencia enormemente de otros lenguajes más populares tales como Fortran, Pascal, C o Java[1].

DEFINICIONES

Prolog (o PROLOG), proveniente del francés PROgrammation en LOGique, es un lenguaje de programación lógico e interpretado usado habitualmente en el campo de la Inteligencia artificial[2].

Prolog es un lenguaje de programación hecho para representar y utilizar el conocimiento que se tiene sobre un determinado dominio. El dominio es un conjunto de objetos y el conocimiento se representa por un conjunto de relaciones que describen las propiedades de los objetos y sus interrelaciones. Un conjunto de reglas que describa estas propiedades y estas relaciones es un programa **Prolog**.

Hechos

Expresan relaciones entre objetos. Supongamos que queremos expresar el hecho de que "una gallina pone huevos". Este hecho, consta de dos objetos, "gallina" y "huevos", y de una relación llamada "pone". La forma de representarlo en **Prolog** es: pone (gallina, huevos).

Los nombres de objetos y relaciones deben comenzar con una letra minúscula.
Primero se escribe la relación, y luego los objetos separados por comas y encerrados entre paréntesis. Al final de un hecho debe ir un punto (el carácter ".").
El orden de los objetos dentro de la relación es arbitrario, pero debemos ser coherentes a lo largo de la base de hechos.

Elaborado por: William A. Abrego V.

Variables

Representan objetos que el mismo **Prolog** determina. Una variable puede estar instanciada o no instanciada. Esta instanciada cuando existe un objeto determinado representado por la variable. De este modo, cuando preguntamos "Una gallina pone X ?", **Prolog** busca en los hechos cosas que pone una gallina y respondería: X = huevos, instanciando la variable X con el objeto huevos.
Los nombres de **variables** comienzan siempre por una letra mayúscula.
Un caso particular es la variable anónima, representada por el carácter subrayado ("_"). Es una especie de comodín que utilizaremos en aquellos lugares que debería aparecer una variable, pero no nos interesa darle un nombre correcto ya que no vamos a utilizarla posteriormente.

Reglas

Las reglas se utilizan en **Prolog** para representar que un hecho depende de uno o más hechos, en otras palabras, una regla sirve para demostrar conocimiento en lenguaje natural.
Se expresa mediante una sentencia condicional. Por ejemplo, si en lenguaje natural decimos «si X es abuelo de Y entonces Y es nieto de X», en Prolog escribiremos:
nieto (Y, X) :- abuelo(X, Y).. Son la representación de las implicaciones lógicas del tipo p⇒q (p implica q).

Una regla consiste en una cabeza y un cuerpo, unidos por el signo ":-", la cabeza está formada por un único hecho, el cuerpo puede ser uno o más hechos (conjunción de hechos), separados por una coma (","), que actúa como el "y" lógico, las reglas finalizan con un punto (".").

La cabeza en una regla **Prolog** corresponde al consecuente de una implicación lógica, y el cuerpo al antecedente. Este hecho puede conducir a errores de representación. Supongamos el siguiente razonamiento lógico: tiempo(lluvioso) −> suelo (mojado). suelo (mojado).

Que el suelo esta mojado, es una condición suficiente de que el tiempo sea lluvioso, pero no necesaria. Por lo tanto, a partir de ese hecho, no podemos deducir mediante la implicación, que esta, lloviendo (pueden haber regado las calles).
La representación correcta en **Prolog**, sería: suelo(mojado):- tiempo(lluvioso).suelo (mojado).

Cabe señalar que la regla esta "al revés". Esto es así por el mecanismo de deducción hacia atrás que emplea **Prolog**. Si cometiéramos el error de representarla como: tiempo(lluvioso): - suelo(mojado). suelo (mojado), **Prolog**, partiendo del hecho de que el suelo esta mojado, deduciría incorrectamente que el tiempo es lluvioso. Para generalizar una relación entre objetos mediante una regla, utilizaremos variables. Por ejemplo:
Representación lógica | Representación PROLOG
Es una gallina(X) —> | pone (X, huevos): pone (X, huevos) | es una gallina(X).

Elaborado por: William A. Abrego V.

Con esta regla generalizamos el hecho de que cualquier objeto que sea una gallina, pondrá huevos. Al igual que antes, el hecho de que un objeto ponga huevos no es una condición suficiente de que sea una gallina. Por lo tanto, la representación inversa sería incorrecta[3].

Operadores

Son predicados predefinidos en **Prolog** para las operaciones matemáticas básicas. Su sintaxis depende de la posición que ocupen, pudiendo ser infijos o prefijos. Ejemplo, el operador suma ("+"), podemos encontrarlo en forma prefija '+(4,8)' o bien infija, '4 + 8'.

También dispone de predicados de igualdad y desigualdad.
X = Y igual
X ≠ Y distinto
X < Y menor
X > Y mayor
X =< Y menor o igual
X >= Y mayor o igual

El operador 'is'.
Es un operador infijo, que en su parte derecha lleva un término que se interpreta como una expresión aritmética, contrastándose con el término de su izquierda.
Por ejemplo, la expresión '9 is 4+7.' es falsa. Por otra parte, si la expresión es 'X is 9+1.', el resultado será la instanciación de X: X = 10[3].

Comandos Básicos

consult.
El predicado "consult" está pensado para leer y compilar un programa **Prolog** o bien para añadir las cláusulas existentes en un determinado fichero a las que ya están almacenadas y compiladas en la base de datos. Su sintaxis puede ser una de las siguientes:
consult(fichero).
consult('fichero.ext').
consult('c:iaprologfichero').

forget.
Tiene como fin eliminar de la base de datos actual aquellos hechos consultados de un fichero determinado. Su sintaxis es: forget(fichero).

exitsys.
Este predicado nos devuelve al sistema operativo.

Entrada/Salida

PROLOG, al igual que la mayoría de los lenguajes de programación modernos incorpora predicados predefinidos para la entrada y salida de datos. Estos son tratados como reglas que siempre se satisfacen.

write. Su sintaxis es: write('Hola Mundo').

Las comillas simples encierran constantes, mientras que todo lo que se encuentra entre comillas dobles es tratado como una lista. También podemos mostrar el valor de una variable, siempre que este instanciada: write(X).

nl. Este predicado fuerza un retorno de carro en la salida. Por ejemplo:

```
write('linea 1'), nl,
write('linea 2').
```

tiene como resultado:

```
linea 1
linea 2
```

read. Lee un valor del teclado. La lectura del comando read no finaliza hasta que se introduce un punto ".". Su sintaxis es: read(X).

Instancia la variable X con el valor leído del teclado.

```
read(hola).
```

Se evalúa como cierta siempre que lo tecleado coincida con la constante entre paréntesis (en este caso hola).

Tipos de datos en Prolog

Symbol

Hay dos tipos de símbolos:

Un grupo de caracteres consecutivos (letras, números y signos de subrayado) que comienzan con un carácter en minúscula

Ejemplo: grande, grande_perro, el_perro_grande.

Un grupo de caracteres consecutivos (letras y números) que comienzan y terminan con dobles comillas ("). Este tipo es útil cuando se quiere comenzar el símbolo con un carácter en mayúscula o si se quieres agregar espacios entre los caracteres del símbolo.

Ejemplo: "pequeño", "pequeño gato"

String

Cualquier grupo de caracteres consecutivos (letras y números) que comience y termine con dobles comillas ("). Es igual a símbolo, pero Prolog los trata de forma distinta.

Ejemplo: "alto", "alto edificio"

Integer

Cualquier número comprendido entre (-32.768 y 32.768). El límite está determinado porque los enteros se almacenan como valores de 16 bits, este límite puede variar según la versión de Prolog.

Ejemplo: 20,-330,1004

Elaborado por: William A. Abrego V.

Real
Cualquier número real en el rango +/- 1E-307 a +/-1E+308. El formato incluye estas opciones: signo, numero, punto decimal, fracción, E(exponente), signo para el exponente, exponente. Ejemplo: 7, 7.3214.

Char
Cualquier carácter de la lista ASCII estándar, posicionado entre dos comillas sencillas (‘). Ejemplos: ‘t’, ‘X’, ‘f’.

CONCLUIÓN

1. Prolog permite describir estructuras de datos, sus relaciones y objetivos por ser un lenguaje declarativo.

2. El proceso por el cual ejecuta las tareas no se establece de forma explícita en el programa, sino que se determina por el proceso de traducción del lenguaje.

3. Prolog tiene una estructura de programación orientada a dar soluciones a problemas del área de aplicación de la Inteligencia Artificial mediante estructuras de programación orientadas a la resolución de problemas de una manera no-clásica.

4. Podemos valorar a Prolog como el lenguaje específico para la programación basada en hechos, predicados y reglas.

5. Intentar desarrollar soluciones similares con búsquedas de objetivos en otros lenguajes tradiciones sería muy extenso y con poca eficacia.

6. No todos los programadores y docentes de Informática conocen este lenguaje, ni su aplicación en Inteligencia Artificial y Sistemas Inteligentes.

7. Prolog, es una herramienta muy efectiva a la hora de buscar soluciones mediante la Programación Lógica y el aprovechamiento de elementos de otras áreas tales como la Matemática Discreta, ayudándonos a encontrar nuevos datos a partir de otros preexistentes.

BIBLIOGRAFÍA

[1] J. R. Ennals, *Beginning Micro-PROLOG*. Harper & Row Pub., 1984.

[2] W. B. Rauch-Hindin, *Aplicaciones de la inteligencia artificial en la actividad empresarial, la ciencia y la industria*. Ediciones Díaz de Santos, 1989.

[3] E. Tuckler, «Temas generales de prolog», *Monografias.com*, 27 de julio de 2000. https://www.monografias.com/trabajos5/prolog/prolog (accedido 23 de febrero de 2022).