El lenguaje de programación Prolog

Presentación para la materia Teoría del Lenguaje

Axel Straminsky
Demian Ferrerio
Martín Paulucci

Facultad de Ingeniería
Universidad de Buenos Aires

9 de Mayo, 2011

Visión General

- Es el lenguaje más famoso del Paradigma Lógico
- Un programa Prolog consiste en un conjunto de sentencias, que pueden ser hechos o reglas.
- Es conversacional. La interacción con el programa consiste en hacerle preguntas al sistema.
- Usado en Inteligencia Artificial y procesamiento de Lenguajes Naturales.

Historia

- Su nombre proviene de la abreviación *PROgramming in LOGic*.
- Fue ideado a principios de los '70 por Colmerauer y Roussel.
- Nació de un proyecto que no tenía como objetivo la implementación de un lenguaje de programación, sino el procesamiento de lenguajes naturales.
- Inicialmente interpretado, hasta que en 1983 se desarrollo un compilador capaz de traducir Prolog a un conjunto de instrucciones de una máquina abstracta denominada Warren Abstract Machine (WAM), y desde entonces Prolog es semi-interpretado.

Declaratividad

- Describe la *lógica* del programa sin explicitar el flujo de ejecución.
  - Lo opuesto a la programación imperativa.
- Permite abstraerse del "cómo" y concentrarse en el "qué" a la hora de escribir programas.
- El paradigma Lógico, al igual que el Funcional, provienen del paradigma Declarativo.
- Programar de esta manera tiende a reducir los errores causados por "efectos colaterales" (side effects).
- Ideal para implementar Computación Paralela

Paradigma Lógico

- Puramente declarativo, es decir, no tiene estructuras de control.
- La lógica matemática es la manera más sencilla de expresar formalmente problemas complejos para el intelecto humano.
- Las resonsabilidades para la ejecución de una tarea están divididas entre el programador, que debe asegurar que el modelo sea lógicamente coherente, y la máquina, que debe resolver el problema de manera eficiente.
  Los lenguajes lógicos usan sobre todo para las siguientes aplicaciones:

- Inteligencia Artificial
- Demostración automática de Teoremas
- Sistemas expertos
- Reconocimiento de Lenguaje Natural

Implementaciones de Prolog

Algunas implementaciones de Prolog son:

- SWI-Prolog : Soporta mulithreating.
- Mercury : Mezcla de mrogramación lógica y funcional.
- Fprolog: Añade lógica difusa.
- Prolog+ : Añade clases y jerarquías de clases.
- LogTalk: añade *POO*.
- λprolog: soporta polimorfismo y programación de alto orden.

Programación en Prolog: Hechos y Términos simples

- Un programa Prolog puro está compuesto únciamente de un conjunto finito de *Clausulas de Horn*. Hay dos tipos de clausulas: *hechos* y *reglas*.
- Un *hecho* define una verdad del programa. Por ejemplo:

```
varon(pedro).
```

  puede leerse "Pedro es un varón".
- "Sólo un tipo de dato": el término
- Existen términos atómicos (e.g. `pedro`, `x`, `'un atomo'`)
- Numéricos (e.g. 6, -15).
- Compuestos
  - Por ejemplo `gusta(martin, jazz)`
    Donde `gusta` es un *functor* (o predicado), que se caracteriza por un nombre y aridad; y se denota `gusta/2`.
    Y `martin` y `jazz` son átomos, que tambien pueden verse como functores de aridad cero.

Programación en Prolog: Variables y Reglas

- Una lista es un caso especial de término compuesto
  e.g. `[1, 2, 3, 5]`
- Una *variable* representa un valor no especificado.
- Sintaxis: los átomos y predicados con minúscula y las variables con mayúscula.
- Una *regla* define una relación del tipo:

$$(p \wedge q \wedge \cdots \wedge t) \Rightarrow u$$

- En Prolog, se escribe primero el consecuente, y después el o los antecedentes. Por ejemplo:

```
hija(A, B) :- mujer(A), madre(B, A).
```

Programación en Prolog: Queries

Las queries son la forma de obtener información del programa.

- Si se especifican todos los parámetros de una consulta con constantes, el intérprete informa si el predicado se cumple o no.

```
?- factorial(5, 120).
true .
```

- Si alguno de los términos de la consulta es una variable, se informa qué valores debe tomar la variable para cumplir ese predicado.

```
?- hermano(zeus, Quien).
Quien = hades ;
Quien = poseidon ;
false.
```

Programación en Prolog: Reglas recursivas

- Ejemplo de regla recursiva:

```
ancestro(A, B) :- padre(A, B).
ancestro(A, B) :- padre(A, X), ancestro(X, B).
```

  - Las condiciones de corte se especifican como hechos.

- Ver ejemplo `ancestro.pl`

Ejemplo: Factorial

- En Prolog no existen instrucciones de control, y su ejecución se basa en 2 conceptos: *unificación* y *backtracking*.
- La unificación produce una ligadura entre 2 términos lógicos que están relacionados mediante una igualdad.
- Veamos un ejemplo:

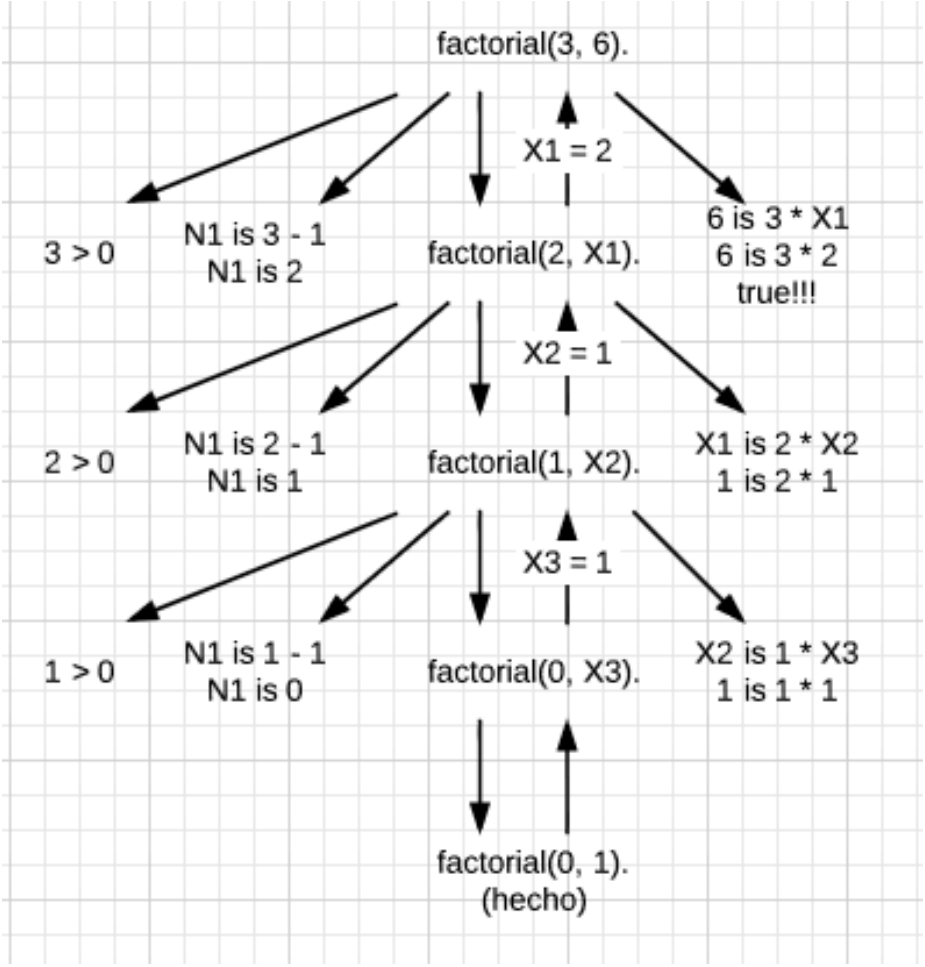
```
factorial(0, 1).
factorial(N, F) :- N > 0, N1 is N - 1,
    factorial(N1, F1), F is N * F1.
```

  - El flujo de control se genera utilizando recursividad.
  - Se usa como función de corte a los hechos.

Backtracking

Ejemplo de seguimiento.

```
?- factorial(3, 6).
true .
```



Ejemplos interesantes

Ver ejemplos `cambio.pl` y `n_reinas.pl`.

Desventajas de Prolog

- Prolog no es un lenguaje de proposito general.
- No tiene un sistema de módulos, lo cual hace que sea poco escalable.
- No hay un estandard bien definido
  - Incompatibilidades entre distintas implementaciones.
- Baja performance en comparación con otros lenguajes.
  - Poco apropiado para programas computacionalmente complejos.

Fin

Preguntas

Fin

Preguntas
Muchas gracias!