Paradigmas de Programação - ELC117

Trabalho 2 - Prolog

Dupla: Mathias Recktenvald e Guilherme Brizzi

Introdução

Prolog é uma linguagem de programação de paradigma lógico, diferentemente das linguagens mais convencionais, que adotam uma abordagem procedural e imperativa (como o C) ou que adicionam elementos de orientação ao objeto (como Java).

Esse paradigma lógico adotado torna a linguagem declarativa e faz com que, em Prolog, o programador defina ***relações*** entre diferentes ***fatos*** e ***regras***. A construção de um programa em Prolog não é feita escrevendo instruções que controlam o fluxo, mas é baseada na formação de inferências e relações lógicas.

Sintaxe e funcionamento

Fatos e Regras

Por exemplo, podemos dizer em linguagem natural humana que *bolo é um doce* e que *pastel é um salgado*. Ambas declarações são ***fatos***. Em Prolog, podemos realizar essa mesma declaração de fatos da seguinte maneira:

```
doce(bolo).
salgado(pastel).
```

Podemos agora definir uma regra lógica que gera relações a partir dos fatos que declaramos. Em linguagem natural, podemos dizer, por exemplo, que doces são comida e que salgados são comida. Em Prolog, traduzimos isso como:

```
comida(X) :- doce(X).
comida(X) :- salgado(X).
```

Queries

A partir do estabelecimento desses fatos e regras, torna-se possível realizar *queries* que retornam informação sobre as relações entre eles. Vamos começar pelo mais simples: vamos checar se um bolo é um doce ou é um salgado:

```
?- doce(bolo).

// output: true

?-salgado(bolo).

// output: false
```

Podemos agora também checar o resultado gerado pela regra que associa todo doce X ser uma comida X e todo salgado X ser uma comida X. Por exemplo, em linguagem natural, *um bolo é um doce, e todo doce é uma comida, então um bolo é uma comida.* Em Prolog, checamos isso fazendo o *query*:

```
?-comida(bolo).

// output: true

?-comida(pastel).

// output: true
```

Exemplo aprofundado

Vamos agora ver um exemplo mais aprofundado para compreender o funcionamento da linguagem Prolog: vamos usar fatos e regras para formar a estrutura de uma família: João é pai de Maria e José, e Ana é mãe de Maria e José, e Júlia é mãe de Ana, portanto, avó de Maria e José. Declaremos esses fatos em Prolog:

```
pai(joao, maria).
pai(joao, jose).
mae(ana, maria).
mae(ana, jose).
```

Agora, vamos estabelecer algumas regras:

```
filho(X, Y) :- pai(Y, X).
filho(X, Y) :- mae(Y, X).
```

Essas regras estabelecem que a pessoa X é filha da Y se Y é pai/mãe da X. Basicamente, estamos criando uma nova regra que conseguirá relacionar os fatos declarados sobre o pai e mãe de cada filho. Podemos fazer então consultas:

```
?-pai(joao, Y).
// equivale a perguntar quem são os filhos Y de João
// output: Y = maria ;
           Y = jose.

?-pai(X, maria).
```

```
// equivale a perguntar quem é o pai de Maria
// output: X = joao;

?-filho(maria, ana).
// equivale a perguntar se Maria é filha de Ana
// output: true
```

Regras recursivas

Em alguns casos, para englobar as possíveis relações lógicas que se busca descrever, podemos ter que utilizar recursão na construção de regras. Partindo do exemplo anterior, podemos pensar em checar se alguém é um ancestral de outro. Ou seja, se alguém é mãe, pai, avô, avó e assim por diante de uma pessoa.

Para isso podemos criar uma nova regra em Prolog:

```
ancestral(X, Y) :- pai(X, Y).
ancestral(X, Y) :- mae(X, Y).
ancestral(X, Y) :- pai(X, Z), ancestral(Z, Y).
ancestral(X, Y) :- mae(X, Z), ancestral(Z, Y).
```

Vamos adicionar um novo fato: Júlia é mãe de Ana e, portanto, é avó de Maria e José. Em Prolog:

```
mae(julia, ana).
```

Para testar, realizamos um query para vermos se Júlia é ancestral de Maria, sua neta:

```
?-ancestral(julia, maria).
// output: true
```

Conclusão

O paradigma de programação lógico traz uma abordagem bastante diferente da programação imperativa e procedural e, assim, pode parecer bastante abstrata e distante. Porém, a linguagem Prolog nunca obteve grande sucesso e é hoje pouco usada, com exceção de algumas aplicações nichadas, como em alguns bancos de dados e programas matemáticos.

Referências Externas

- https://www.ime.usp.br/~slago/slago-prolog.pdf
- https://www.swi-prolog.org/
- https://www.geeksforgeeks.org/prolog-an-introduction/