Programação Lógica

Prolog: alteração da base de dados, predicados que fornecem todas as soluções, exemplos

Profa Heloisa de Arruda Camargo

Priscilla de Abreu Lopes (PESCD)
priscilla_lopes@dc.ufscar.br

Alteração da Base de Dados

- Alguns predicados do Prolog modificam a base de dados em tempo de execução
- ***abolish(Pred)*** apaga todos os predicados especificados pelo seu argumento
  - » +Pred especificação de predicado

Alteração da Base de Dados

- Ex: Base de dados:

```
estudante('Joao').
estudante('Marcia').
estudante('Paulo').
faltas('Joao',5).
faltas('Marcia,2).
```

?- abolish(faltas/2).

?- abolish([estudante/1, faltas/2]).

Alteração da Base de Dados

- ***abolish(Pred,Arid)***

+Pred	**funtor**
+Arid	**número de argumentos**

- **?- abolish(faltas,2).**

Alteração da Base de Dados

- *assert(Claus)* adiciona uma cláusula no fim das cláusulas associadas com seu predicado

  +Claus cláusula (fato ou regra)
- ?- assert(gosta(maria,cinema)).
- ?- assert((gosta(ana,Pessoa) :- gosta(Pessoa, computacao))).

Alteração da Base de Dados

- Obs:
  - as regras devem aparecer entre parêntesis.
  - variáveis não instanciadas no momento da execução do assert são consideradas variáveis.

Alteração da Base de Dados

- *assert(Claus, Pos)* adiciona uma cláusula na posição especificada

  +Claus cláusula (fato ou regra)

  +Pos inteiro >= 0
- *asserta(Claus)* adiciona uma cláusula no começo das cláusulas associadas

Alteração da Base de Dados

- assertz(Claus) adiciona uma cláusula no fim das cláusulas associadas
- retract(Claus) Procura a primeira cláusula na base de dados que unifica com Claus, se encontrar apaga. Variáveis são instanciadas. Permite backtracking.
  - +Claus cláusula

Exemplo retract(Claus)

Ex:- Base de dados

```
estudante('Joao').
estudante('Marcia').
estudante('Paulo').
faltas('Joao',5).
faltas('Marcia,2).
Aceita(Nome, Curso) :- estudante(Nome),
                       faltas(Nome,N),
                       N < 5.
```

Exemplo retract(Claus)

?- retract(estudante(X)).

X='Joao';

?- retract((aceita(N,C) :- B, D, E)).

N = _

C = _

B=estudante(Nome)

D=faltas(Nome,N),

E= N < 5

Predicados que fornecem todas as soluções

- **São predicados de Segunda ordem.**
- **Encontram todas as soluções possíveis para um objetivo, e as retorna em forma de lista.**

Predicados que fornecem todas as soluções

- **bagof(Termo, Objetivo, Lista)**
  - ?Termo - qualquer termo do Prolog
  - +Objetivo - uma cláusula na forma de consulta
  - ?Lista - variável
- **Retorna sucesso se Lista é uma lista não vazia de instâncias de Termo, que tornam Objetivo verdadeiro.**
- **Falha se não houverem soluções para o Objetivo.**

Exemplos

- Base de dados:

```
joga(andre,volei).
joga(andre,futebol).
joga(paulo,futebol).
joga(monica,futebol).
joga(adriana,volei).
joga(maria,basket).
joga(carlos,basket).
```

Exemplos

- Consultas:

```
?- joga(Quem,futebol).
Quem = andre ;
Quem = paulo ;
Quem = monica ;
No
```

Exemplos

```
?- bagof(X,joga(X,futebol),Lista).
X = _ ,
Lista = [andre,paulo,monica]
```

- O primeiro argumento (Termo) pode ser qualquer termo:

```
?- bagof(achou(X), joga(X,futebol),Lista).
X = _ ,
Lista = [achou(andre),achou(paulo),achou(monica)]
```

Exemplos

- Se não hoverem respostas, falha

```
?- bagof(X,joga(X,tenis),Lista).
no
```

- Mantem respostas repetidas e desordenadas:

```
?- bagof(X,pertence(X,[a,b,c,a,d,e,a,c,a]),S).
X = _ ,
S = [a,b,c,a,d,e,a,c,a]
```

Exemplos

- O objetivo pode ser composto:

  ?- bagof(X,(joga(X,futebol),joga(X,volei)),L).

  X = _ ,

  L = [andre]

  ?- bagof(X,(pertence(X,[1,2,3,4,5]),X<4),L).

  X = _ ,

  L = [1,2,3]

Exemplos

- **Permite o uso de variáveis livres (aparecem no objetivo, e não aparecem no termo):**

  ?- bagof(X,joga(X,Y),Lista).

  X = _ ,

  Y = basket ,

  Lista = [maria,carlos] ;

  X = _ ,

  Y = futebol ,

  Lista = [andre,paulo,monica] ;

  X = _ ,

  Y = volei ,

  Lista = [andre,adriana] ;

  no

Predicados que fornecem todas as soluções

- **setof(Termo, Objetivo, Lista)**

  ?Termo - qualquer termo do Prolog

  +Objetivo - uma cláusula na forma de consulta

  ?Lista - variável
- **Retorna sucesso se Lista é uma lista não vazia de instâncias de Termo, que tornam Objetivo verdadeiro.**
- **Falha se não houverem soluções para o Objetivo.**
- **Diferença do bagof: as respostas são ordenadas e as duplicações são removidas.**

Exemplos

?- setof(X,pertence(X,[a,z,c,a,e,d,a,c,a]),S).

X = _ ,

S = [a,c,d,e,z]

?- bagof(X,pertence(X,[a,z,c,a,e,d,a,c,a]),S).

X = _ ,

S = [a,z,c,a,e,d,a,c,a]

Exemplos

?- bagof(X,joga(X,futebol),Lista).
X = _ ,
Lista = [andre,paulo,monica]

?- setof(X,joga(X,futebol),Lista).
X = _ ,
Lista = [andre,monica,paulo]

Exemplos

?- setof(nada,joga(X,volei),Lista).
X = adriana ,
Lista = [nada] ;
X = andre ,
Lista = [nada] ;
no

Predicados que fornecem todas as soluções

- **findall(Termo, Objetivo, Lista)**
  ?Termo - qualquer termo do Prolog
  +Objetivo - uma cláusula na forma de consulta
  ?Lista - variável
- **Retorna sucesso se Lista é uma lista de instâncias de Termo, que tornam Objetivo verdadeiro.**
- **Se não houverem soluções para o Objetivo, retorna LISTA VAZIA.**
- **As respostas NÃO são ordenadas e as duplicações são MANTIDAS.**

Exemplos

?- findall(X,joga(X,tenis),Lista).
X = _ ,
Lista = [ ]

?- findall(X,joga(X,futebol),Lista).
X = _ ,
Lista = [andre,paulo,monica]

Exemplos

- **Para variáveis livres, constrói uma única solução para todos os possíveis valores das variáveis livres.**

  ?- findall(X,joga(X,Y),Lista).
  X = _ ,
  Y = _ ,
  Lista = [andre,andre,paulo,monica,adriana,maria,carlos]

  ?- findall(X,(pertence(X,[1,3,5,7,3,5]),X>4),Numeros).
  X = _ ,
  Numeros = [5,7,5]