Apontamentos PLR

From Mywiki

Isto são apontamentos pessoais, por isso está bastante *raw* (foi feito para eu perceber). O *disclaimer* do costume: Se algo tiver errado avisem, mas se tiverem coisas erradas no exame por erros aqui presentes, a responsabilidade e' vossa.

Contents

Operadores de Comparação

- = Unifica (negação \=)
- =:= Comparação aritmética (negação =\=)
- == Sintaticamente iguais (negação \==)

O @ como prefixo de <, >, =<, >= estende a encadernador a qualquer termo

Outros Operadores

- functor =..lista

```
f(a,v) =.. [f,a,v].
1+1 =.. [+,1,2].
```

Predicados Importantes

- length(?List,?N)
- member(?Elem,?List)
- functor(Termo,Functor,N)

```
functor(T,f,3), T = f(_A,_B,_C).
```

Cut

Quando se encontra um cut automaticamente se corta o backtracking para vizinhos e fixa-se o que foi unificado à esquerda do cut

Definição de Operadores

```
:- op(prec,assoc,lexeme).
```

Onde a precedência e' um inteiro entre 0 e 1200 e quanto **menor o número maior a precedência**. A associatividade é definida por

In the case of + this atom is yfx, which says that + is an infix operator f represents the operator and x and y the arguments. Furthermore, x stands for an argument which has a precedence which is lower than the precedence of + and y stands for an argument which has a precedence which lower or equal to the precedence of +. There are the following possibilities for what Type may look like:

infix	xfx (non-associative), xfy (right to left), yfx (left to right)
prefix	fx (non-associative), fy (left to right)
suffix	xf (non-associative), yf (right to left)

Eg.

```
:- op(500, xf, is_dead).
```

Ref: [1] (http://cs.union.edu/~striegnk/learn-prolog-now/html/node84.html)

Semântica do operador

A semântica é definida pelo predicado com o mesmo nome que o operador (e com

a mesma ariedade).

Listas por compreensão

- findall - ordem de descoberta
- bagof - igual ao findall mas ordenado
- setof - Igual ao bagof mas sem elementos repetidos

Listas abertas

Ver [2] (http://bulba.sdsu.edu/prolog/diff_list/diff_list.htm)

DCG

Palindrome

```
pal --> [].
pal --> [_].
pal --> [X], cap, [X].
bool(L,N) :- length(L,W), P is floor(2**(W-1)),
        phrase(r(P,N),L).

r(P,N) --> { HP is floor(P/2) }, [0], r(HP,N).
r(P,N) --> { HP is floor(P/2) }, [1], r(HP,NN), {N is P+NN }.
r(0,0) --> [].
```

Restrições

Programa com Restrições

```
:- use_module(library(clpfd)).

send_money([S,E,N,D,M,O,R,Y]) :- L=[S,E,N,D,M,O,R,Y],
                            domain(L,0,9),
                            check(L),
                            labeling([ff],L).

check([S,E,N,D,M,O,R,Y]) :- M  #= 1,
                            Y  #= (E+D) mod 10,
                            CU #= (E+D)/10,
                            E  #= (CU+N+R) mod 10,
                            CD #= (CU+N+R)/10,
                            N  #= (CD+O+E) mod 10,
                            CC #= (CD+O+E)/10,
                            O  #= (S+M+CC) mod 10,
                            CM #= (S+M+CC)/10,
                            M  #= CM,
                            all_distinct([S,E,N,D,M,O,R,Y]).
```

Programa com Restrições que minimiza

```
:- use_module(library(clpfd)).

tasks([A,B,C,D,E],T) :- L=[A,B,C,D,E],
                        domain(L,1,5),
                        check_tasks(L,T),
                        minimize(labeling([],L),T).

% check_tasks(L,T) :- ...
%                     T #= ...
```

Consistência

Consistência de intervalo (Bounds consistency)

Corta intervalos baseado nos operadores =, >, <, ...

Consistência de arco

1. Verificar consistência de nós.
2. Para cada valor possível x para uma variável X verificar que se X=x então é possível satisfazer as outras variáveis que têm restrições

relacionadas com X (i.e. que são sucessores do nó X no grafo de restrições).

Consistência de nó

Remover restrições unirias. ie. no caso de dom(X)={1,2,4,8} e houver a restricao X != 4 entao dom(X)={1,2,8}.

Deve-se propagar as restrições de atribuições, o nó da atribuição desaparece dando origem a laços que se apagam cortando os elementos do domino.

Eg. [3] (http://img188.imageshack.us/img188/3895/exameplr2008alinea2b.png)

Consistência de caminho

Igual à consistência de arcos mas onde a verificação 2 se estende a **todas** as variáveis (e não apenas às que são sucessoras no grafo de restrições)

TODO

- asserts