Listes en Prolog

Listes : arrêt à la fin

```
% longueur(L,N)

longueur([],0).

longueur([X|L],N) :-
        longueur(L,N1),
        N is N1+1.
```

Listes : arrêt à un élément

```
% membre(X,L)

membre(X,[X|_L]).

membre(X,[_Y|L]) :-
    membre(X,L).
```

Listes : arrêt à une position

```
% nieme(N,L,X)

nieme(1,[X|_L],X).

nieme(N,[_X|L],Elt) :-
    N1 is N-1,
    nieme(N1,L,Elt).
```

Récursivité multiple

?-membre_rec(b,[a,[b,c]]).

Récursivité multiple

```
membre_rec(X,[X|_L]).

membre_rec(X,[Tete|Reste]) :-
    atom(Tete),
    membre_rec(X,Reste).

membre_rec(X,[Tete|Reste]) :-
    not(atom(Tete)),
    (membre_rec(X,Tete);membre_rec(X,Reste)).
```

Type des termes

var
nonvar
integer
float
atom
number
atomic
string
ground

Quicksort (Hoare)

```
% quicksort(L,LTriee)

quicksort([],[]).

quicksort([X|L],Triee) :-
    partition(L,X,Petits,Grands),
    quicksort(Petits,PetitsTries),
    quicksort(Grands,GrandsTries),
    append(PetitsTries,[X|GrandsTries],Triee).
```

Quicksort

```
% partition(L,Y,Petits,Grands)

partition([],Y,[],[]).

partition([X|L],Y,[X|Petits],Grands) :-
    X<Y, partition(L,Y,Petits,Grands).

partition([X|L],Y,Petits,[X|Grands]) :-
    X>=Y, partition(L,Y,Petits,Grands).
```

Machines should work,
people should think.

Richard Hamming

Prédicats extralogiques

Pourquoi contrôler l'exécution ?

```
% min(X,Y,Min)

min(X,Y,X) :- X=<Y.

min(X,Y,Y) :- X>Y.
```

Pourquoi contrôler l'exécution ?

```
% membre(X,L)

membre(X,[X|_L]).

membre(X,[_Y|L]) :-
    membre(X,L).
```

réussit autant de fois que l'élément a d'occurrences.

Problème

Ce contrôle est extralogique : il ne peut s’exprimer en logique d’ordre 1.

La coupure !

Donne une solution immédiate pour les deux exemples précédents :

```
min(X,Y,X) :- X=<Y, !.

min(X,Y,Y) :- X>Y.
```

La coupure !

Donne une solution immédiate pour les deux exemples précédents :

```
membre(X,[X|_L]) :- !.

membre(X,[_Y|L]) :-
    membre(X,L).
```

Fonctionnement général

p :- …

p :- …

p :- B_1, B_2, …, B_k, !, B_{k+1}, …, B_n.

p :- …

p :- …

Valeur logique

Le cut est évalué à vrai.

Effets extralogiques

p :- …

p :- …

p :- B_1, B_2, …, B_k, !, B_{k+1}, …, B_n.

p :- …

p :- …

Le backtrack est bloqué sur les B_1, B_2, …, B_k.

Effets extralogiques

Les substitutions des B_{k+1}, …, B_n doivent <u>compléter</u> celles des B_1, B_2, …, B_k.

Effets extralogiques

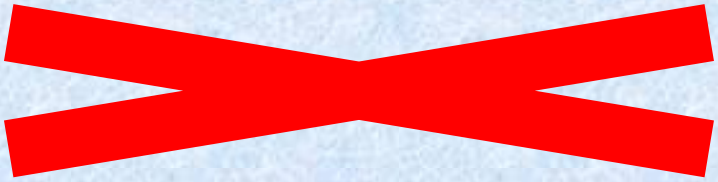
p :- ...

p :- ...

p :- $B_1, B_2, \ldots, B_k, !, B_{k+1}, \ldots, B_n$.

p :-

p :-



Les autres clauses sont ignorées.

Sémantique

On a donc défini un ‘si-alors-sinon’ à utiliser **avec les plus extrêmes précautions.**

p :- …
p :- …
p :- $B_1, B_2, \ldots, B_k, !, B_{k+1}, \ldots, B_n.$
p :- …
p :- …

si **alors**

sinon

Prédicat different

```
different(X,X) :- !, fail.

different(X,Y).
```

Négation

```
not(But) :- call(But), !, fail.

not(But).
```

Pour insister sur le fait qu'on a pas une négation standard, il est recommandé d'écrire \+ (\ pour non et + pour prouvable).