Spécifications formelles – Programmation logique et Prolog

cours 2

Guillaume Petiot

guillaume.petiot@cea.fr

CEA, LIST, LSL

2014-2015

Rappels

Précédemment :

- structure d'un programme Prolog
- sémantique des règles :

```
grandparent(X,Y) :-
    parent(X,P), parent(P,Y).
```

$$\forall X, Y, P.((parent(X, P) \wedge parent(P, Y)) \implies grandparent(X, Y))$$

- unification
- points de choix et "backtracking"
- listes

Unification

Différents opérateurs d'égalité

`X = Y` `X \= Y`	`X == Y` `X \== Y`
sont-ils unifiables ?	sont-ils liés à la même valeur ?
unification	pas d'unification

```
my_append(L1,L2,Res) :- %% Cas 1 : liste vide
    L1 = [], %% unification de L1 et []
    Res = L2. %% unification de Res et L2

my_append(L1,L2,Res) :- %% Cas 2 : liste non vide
    L1 = [H|T], %% unification de L1 et [H|T]
    my_append(T,L2,R),
    Res = [H|R]. %% unification de Res et [H|R]
```

Opérations arithmétiques

- `-X` : moins unaire
- `X+Y` : addition
- `X-Y` : soustraction
- `X*Y` : multiplication
- `X/Y` : division
- `X//Y` : division entière
- `X mod Y` : reste de la division entière
- `X^Y` : exponentiation
- `abs(X)` : valeur absolue

Évaluation arithmétique

```
inc(X,N) :- N = X+1.
```

```
?- inc(5,N).
```

Évaluation arithmétique

```
inc(X,N) :- N = X+1.
```

```
?- inc(5,N).
```

```
N = 5+1.
```

- construction d'un terme composé
- pas d'évaluation

Évaluation arithmétique

```
inc(X,N) :- N = X+1.
```

```
?- inc(5,N).
```

```
N = 5+1.
```

- construction d'un terme composé
- pas d'évaluation

```
inc(X,N) :- N is X+1.
```

```
?- inc(5,N).
N = 6.
```

- `X+1` doit être un terme clos (*ground term*)
- unification de `N` avec l'évaluation de `X+1`

Exemple : Taille d'une liste

```
%% len(Liste, Taille).
```

Exemple : Taille d'une liste

```
%% len(Liste, Taille).

len([],0). %% Cas 1 : liste vide

len([H|T], N) :- %% Cas 2 : liste non vide
    len(T,M),
    N is M+1.
```

Comparaisons arithmétiques

- `X =:= Y`
- `X =\= Y`
- `X < Y`
- `X =< Y`
- `X > Y`
- `X >= Y`

`X` et `Y` doivent être liées à des expressions arithmétiques closes

Exemple : Recherche de maximum dans une liste

```
%% max_of_list(Liste, Maximum).
%% Le maximum d une liste vide est indefini.
```

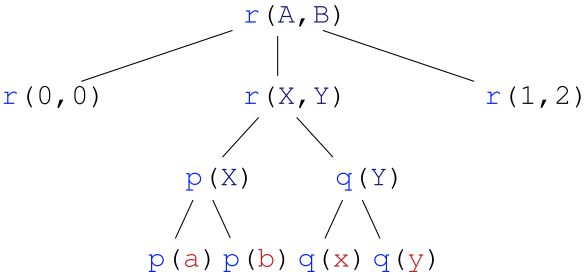
Exemple : Recherche de maximum dans une liste

```
%% max_of_list(Liste, Maximum).
%% Le maximum d une liste vide est indefini.

max_of_list([X],X). %% Cas 1 : 1 element

max_of_list([H|T], Max) :- %% Cas 2 : N>1 elements
    max_of_list(T,M), %% 1er element > Max
    H > M,
    Max = H.

max_of_list([H|T], Max) :- %% Cas 3 : N>1 elements
    max_of_list(T,M), %% 1er element <= Max
    H =< M,
    Max = M.
```

Rappel : Backtracking

```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```

Arbre de recherche :

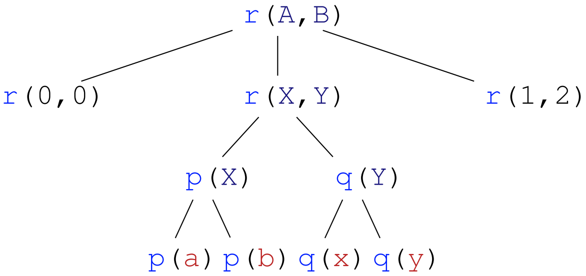


Solutions :

Rappel : Backtracking

```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```

Arbre de recherche :

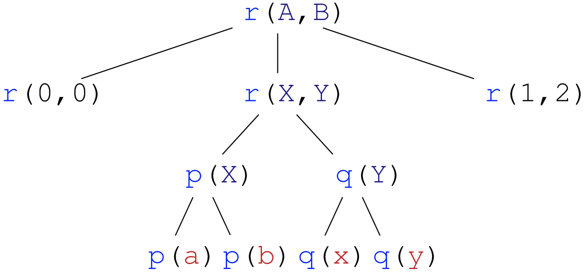


Solutions : `A = B,B = 0`

Rappel : Backtracking

```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```

Arbre de recherche :

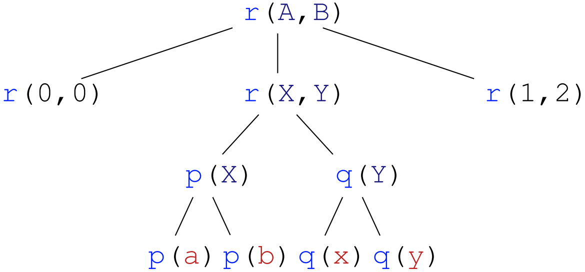


Solutions : `A = B,B = 0 ; A = a,B = x`

Rappel : Backtracking

```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```

Arbre de recherche :

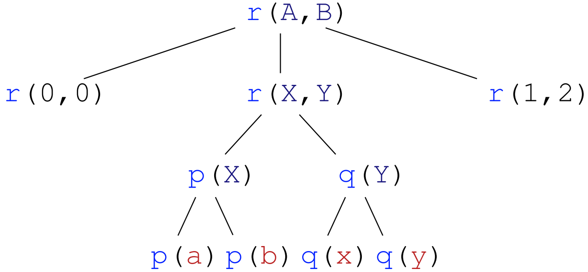


Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y`

Rappel : Backtracking

```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```

Arbre de recherche :

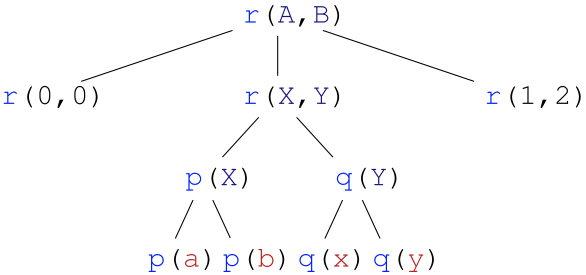


Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x`

Rappel : Backtracking

```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```

Arbre de recherche :

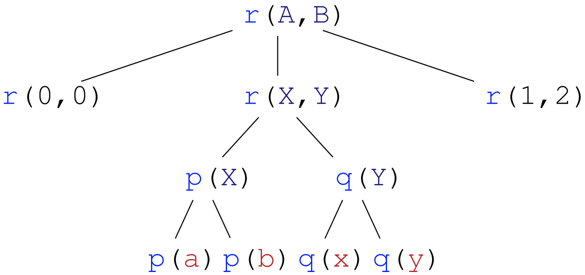


Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y`

Rappel : Backtracking

```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```

Arbre de recherche :

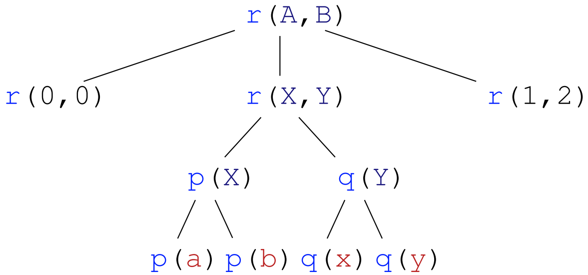


Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y ; A = 1,B = 2`

Rappel : Backtracking

```prolog
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```

Arbre de recherche :



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y ; A = 1,B = 2 .`

Opérateur de coupure

! (“ Cut ”)

```
q :- p1, ...,pn, !, r1, ...,rm
```

Définition :

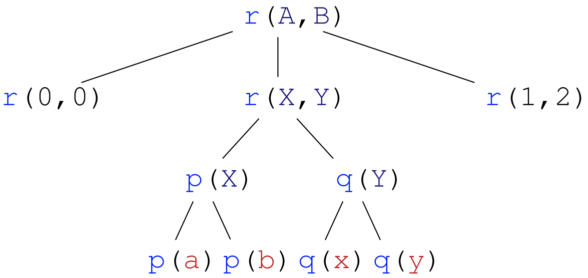
- n’explore pas les autres choix pour les buts `p1`, ..., `pn`
- ignore les clauses suivantes pour `q`
- on peut toujours backtracker sur les buts `r1`, ..., `rm`

Caractéristiques :

- davantage de contrôle sur le backtracking
- peut rendre la résolution plus efficace
- à manier avec précaution

Cut – exemple 1

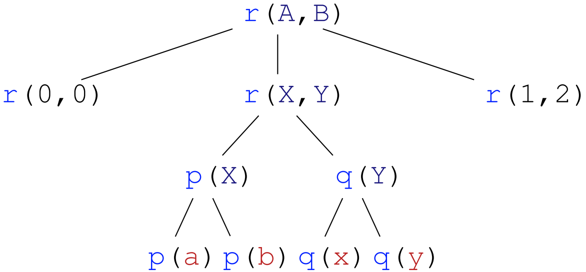
```prolog
%% Exemple :
p(a) :- !.
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions :

Cut – exemple 1

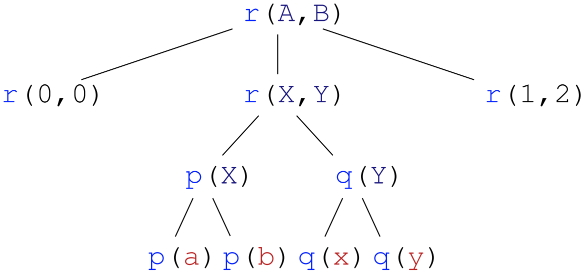
```
%% Exemple :
p(a) :- !.
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0`

Cut – exemple 1

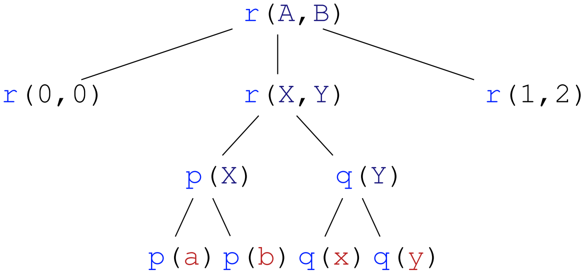
```
%% Exemple :
p(a) :- !.
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x`

Cut – exemple 1

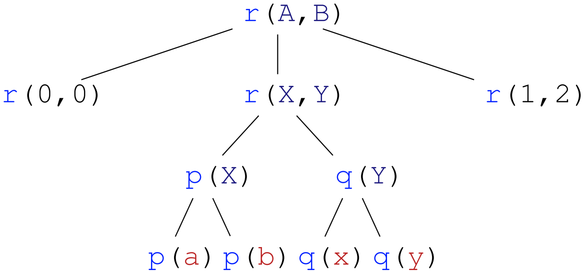
```
%% Exemple :
p(a) :- !.
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y`

Cut – exemple 1

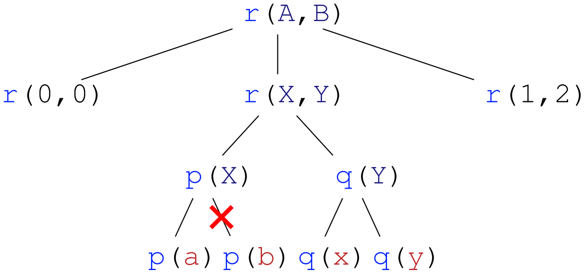
```
%% Exemple :
p(a) :- !.
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = 1,B = 2`

Cut – exemple 1

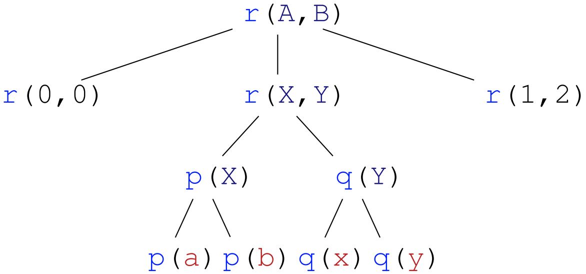
```
%% Exemple :
p(a) :- !.
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = 1,B = 2 .`

Cut – exemple 2

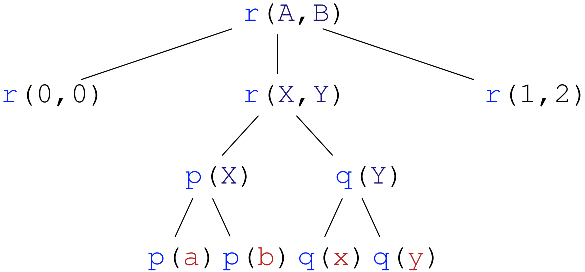
```
%% Exemple :
p(a).
p(b) :- !.
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions :

Cut – exemple 2

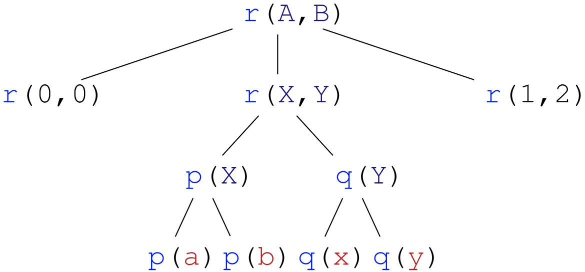
```prolog
%% Exemple :
p(a).
p(b) :- !.
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0`

Cut – exemple 2

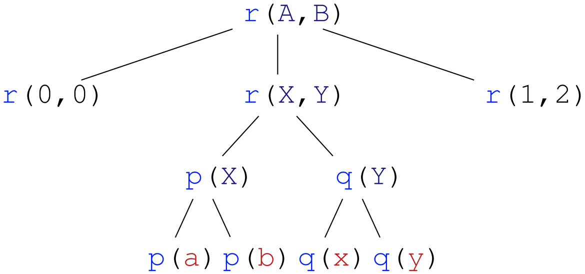
```
%% Exemple :
p(a).
p(b) :- !.
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x`

Cut – exemple 2

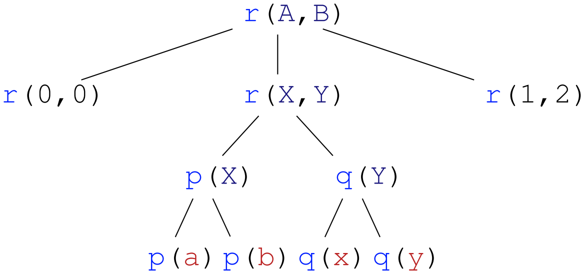
```prolog
%% Exemple :
p(a).
p(b) :- !.
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y`

Cut – exemple 2

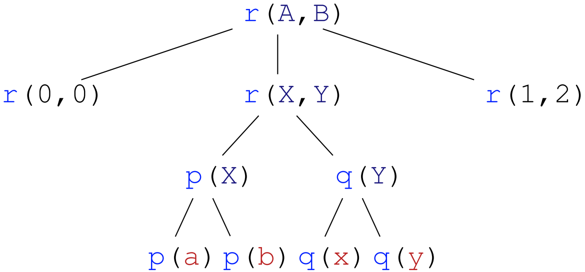
```prolog
%% Exemple :
p(a).
p(b) :- !.
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x`

Cut – exemple 2

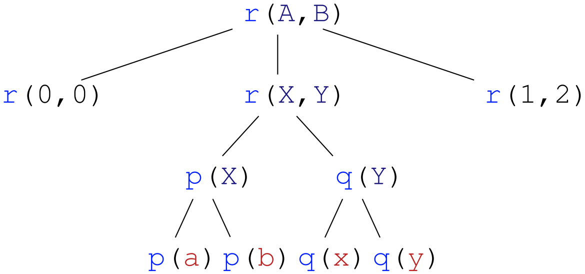
```
%% Exemple :
p(a).
p(b) :- !.
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y`

Cut – exemple 2

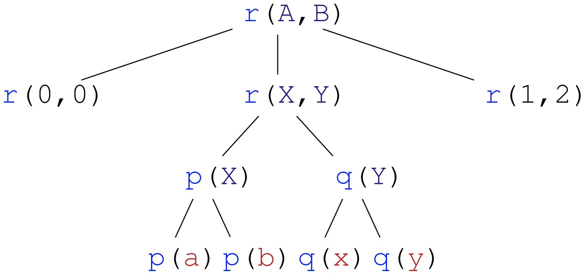
```
%% Exemple :
p(a).
p(b) :- !.
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y ; A = 1,B = 2`

Cut – exemple 2

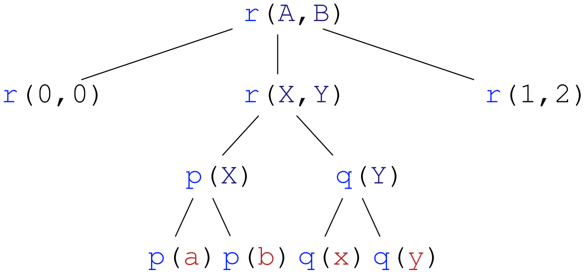
```
%% Exemple :
p(a).
p(b) :- !.
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y ; A = 1,B = 2 .`

Cut – exemple 3

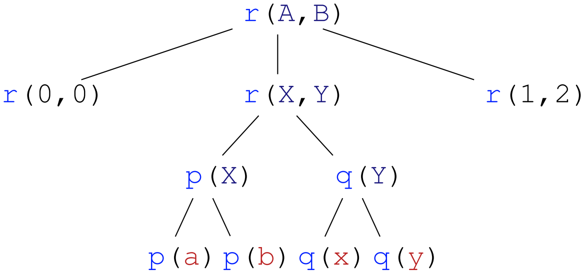
```
%% Exemple :
p(a).
p(b).
q(x) :- !.
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions :

Cut – exemple 3

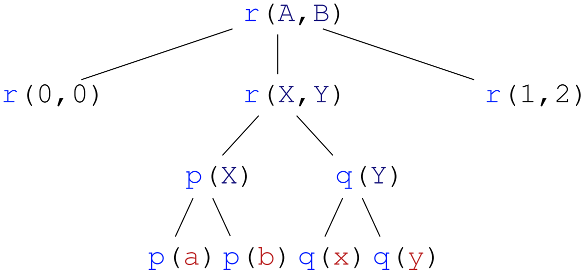
```
%% Exemple :
p(a).
p(b).
q(x) :- !.
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0`

Cut – exemple 3

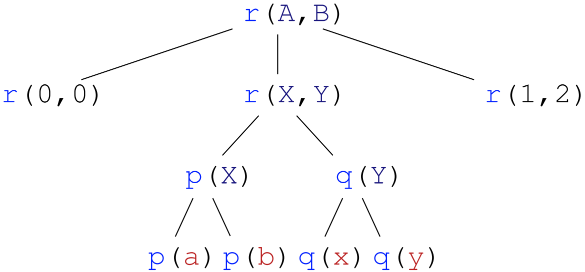
```
%% Exemple :
p(a).
p(b).
q(x) :- !.
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x`

Cut – exemple 3

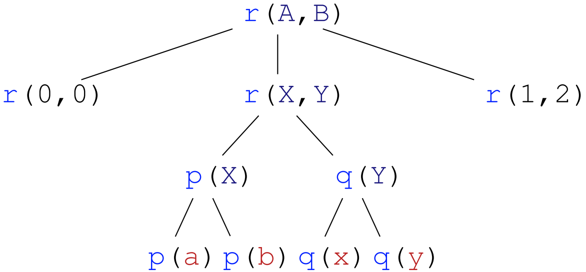
```
%% Exemple :
p(a).
p(b).
q(x) :- !.
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = b,B = x`

Cut – exemple 3

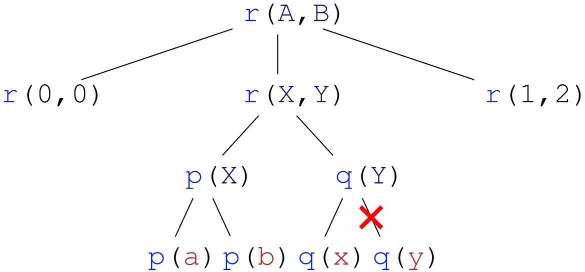
```
%% Exemple :
p(a).
p(b).
q(x) :- !.
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = b,B = x ; A = 1,B = 2`

Cut – exemple 3

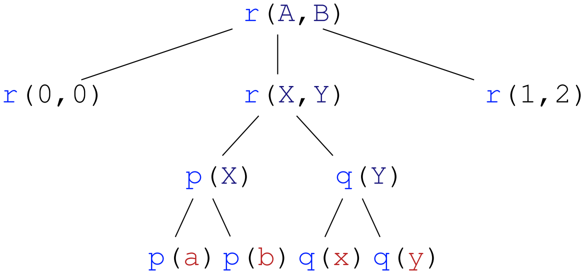
```
%% Exemple :
p(a).
p(b).
q(x) :- !.
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = b,B = x ; A = 1,B = 2 .`

Cut – exemple 4

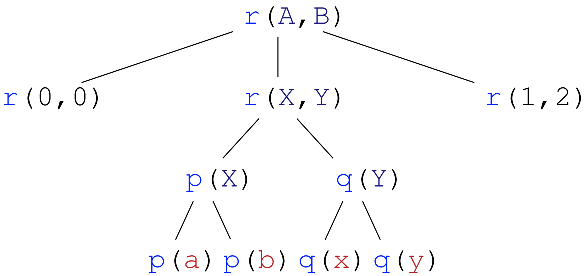
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0) :- !.
r(X,Y) :-
    p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions :

Cut – exemple 4

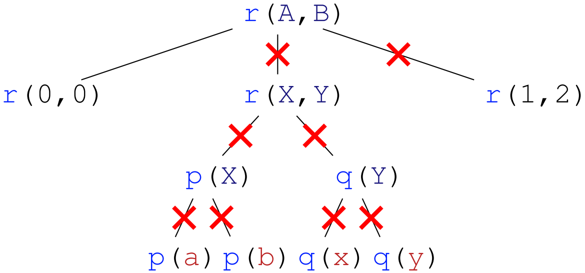
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0) :- !.
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0`

Cut – exemple 4

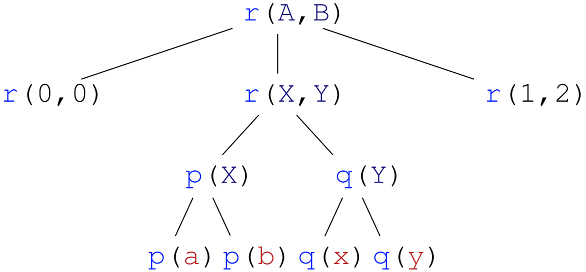
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0) :- !.
r(X,Y) :-
   p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 .`

Cut – exemple 5

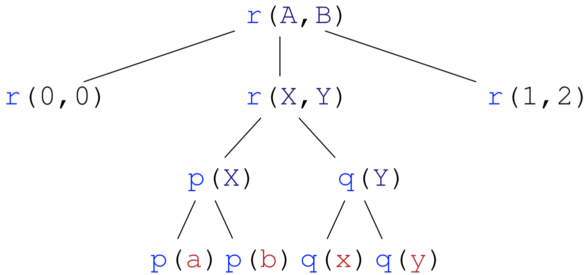
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    !, p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions :

Cut – exemple 5

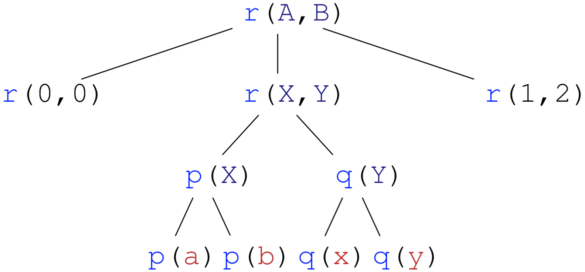
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    !, p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0`

Cut – exemple 5

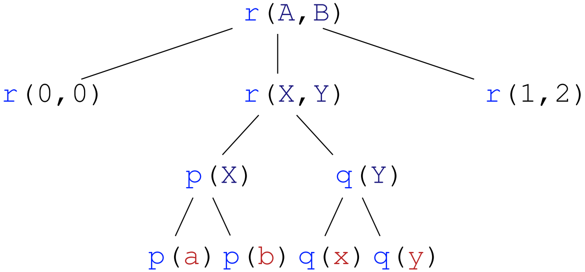
```prolog
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    !, p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x`

Cut – exemple 5

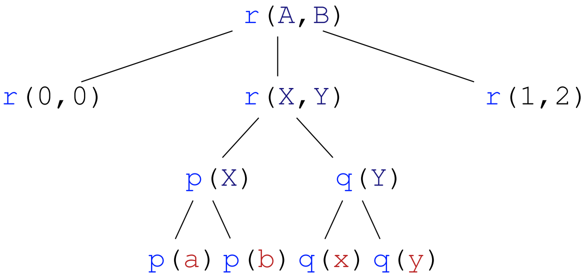
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   !, p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y`

Cut – exemple 5

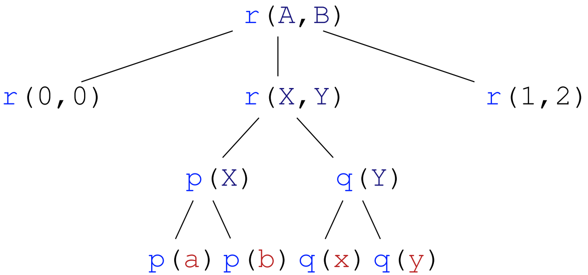
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    !, p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x`

Cut – exemple 5

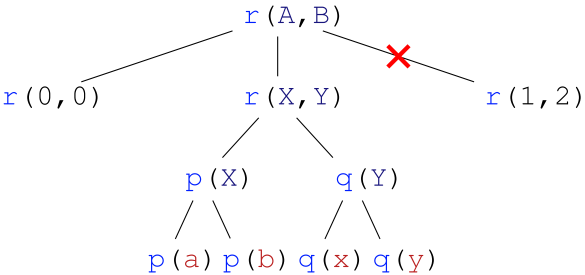
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   !, p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y`

Cut – exemple 5

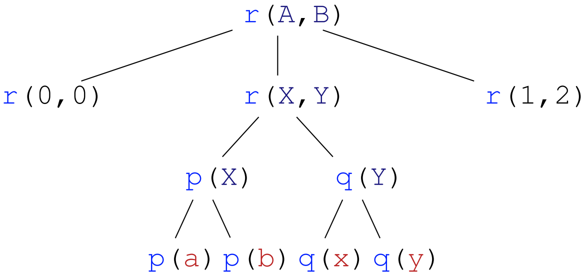
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    !, p(X), q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y .`

Cut – exemple 6

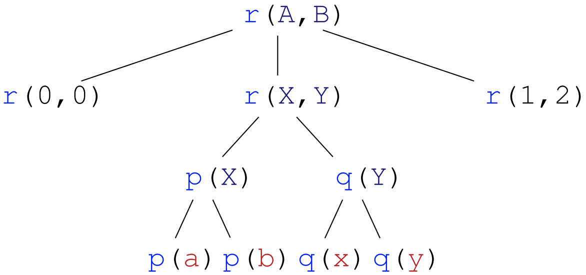
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), !, q(Y).
r(1,2).

?- r(A,B).
```



Solutions :

Cut – exemple 6

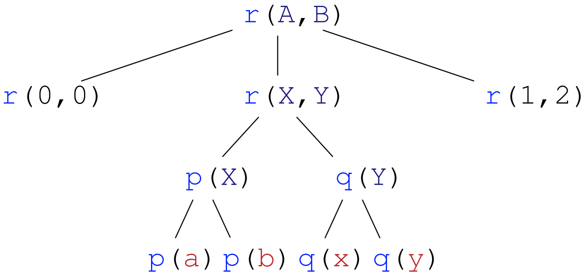
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), !, q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0`

Cut – exemple 6

```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), !, q(Y).
r(1,2).
```

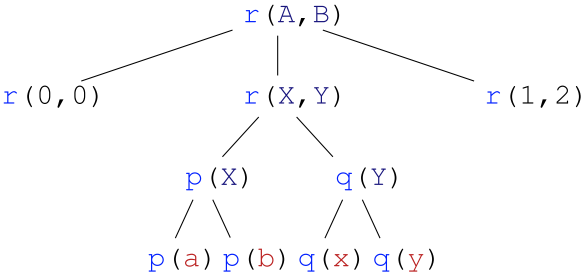



```
?- r(A,B).
```

Solutions : `A = B,B = 0 ; A = a,B = x`

Cut – exemple 6

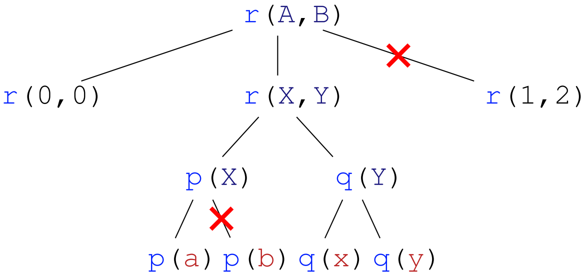
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), !, q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y`

Cut – exemple 6

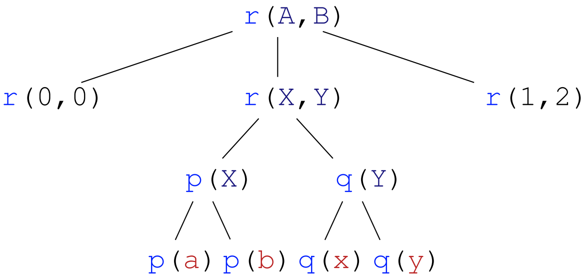
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), !, q(Y).
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y`
.

Cut – exemple 7

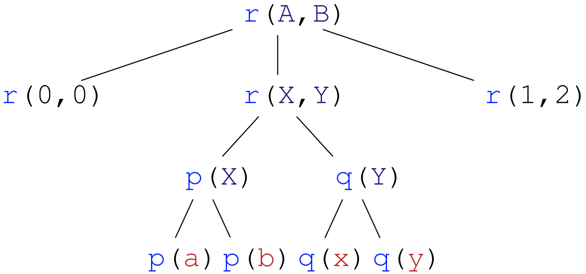
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y), !.
r(1,2).

?- r(A,B).
```



Solutions :

Cut – exemple 7

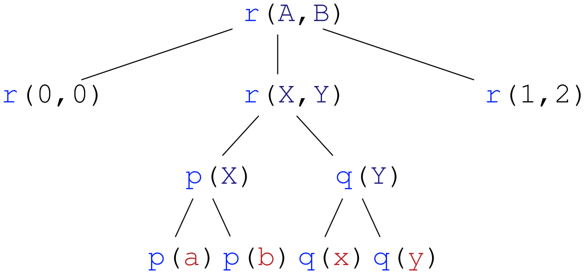
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y), !.
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0`

Cut – exemple 7

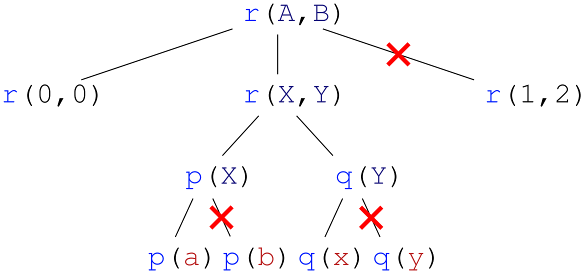
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y), !.
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x`

Cut – exemple 7

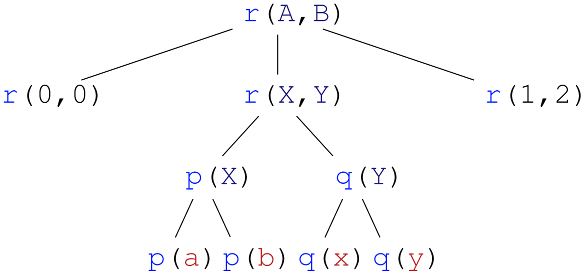
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y), !.
r(1,2).

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x .`

Cut – exemple 8

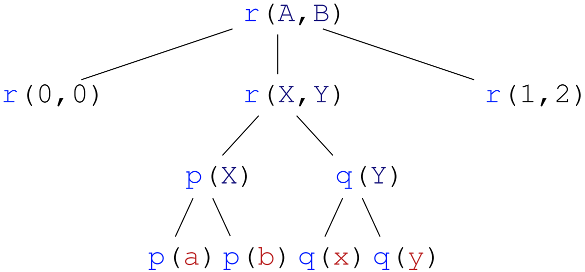
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2) :- !.

?- r(A,B).
```



Solutions :

Cut – exemple 8

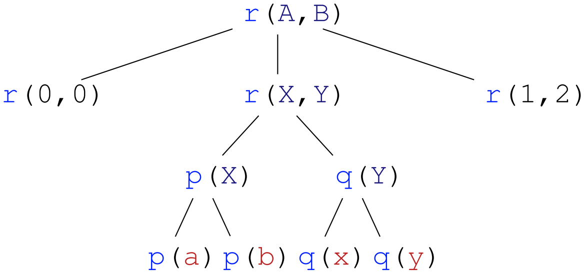
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2) :- !.

?- r(A,B).
```



Solutions : `A = B,B = 0`

Cut – exemple 8

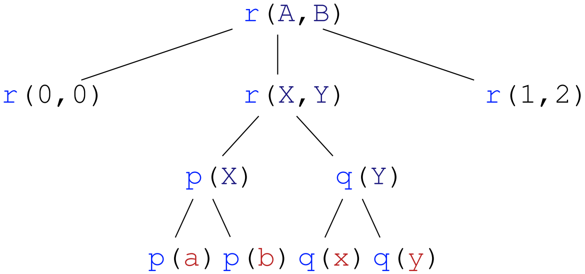
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2) :- !.

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x`

Cut – exemple 8

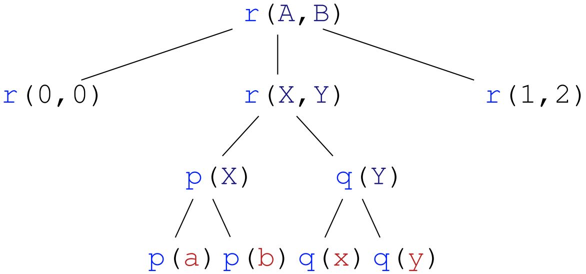
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2) :- !.

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y`

Cut – exemple 8

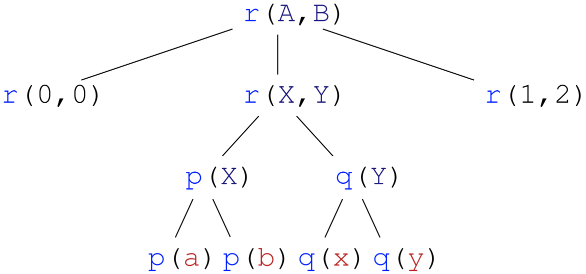
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2) :- !.

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x`

Cut – exemple 8

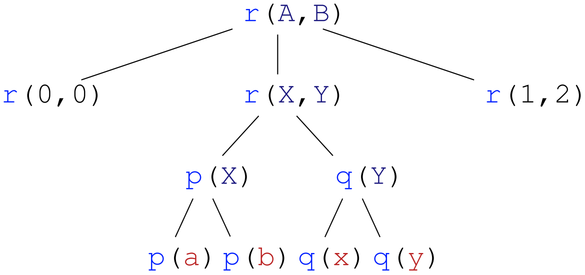
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2) :- !.

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y`

Cut – exemple 8

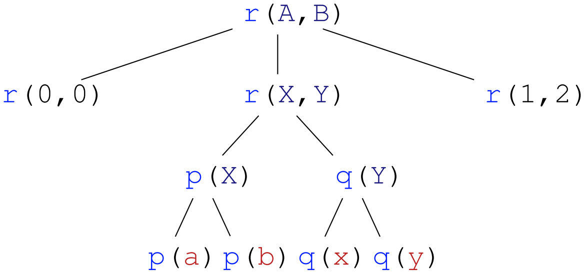
```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
    p(X), q(Y).
r(1,2) :- !.

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y ; A = 1,B = 2`

Cut – exemple 8

```
%% Exemple :
p(a).
p(b).
q(x).
q(y).
r(0,0).
r(X,Y) :-
   p(X), q(Y).
r(1,2) :- !.

?- r(A,B).
```



Solutions : `A = B,B = 0 ; A = a,B = x ; A = a,B = y ; A = b,B = x ; A = b,B = y ; A = 1,B = 2 .`

À retenir sur le cut

```
q :- p1, ...,pn, !, r1, ...,rm
```

- n'explore pas les autres choix pour les buts `p1`, ..., `pn`
- ignore les clauses suivantes pour `q`
- on peut toujours backtracker sur les buts `r1`, ..., `rm`
- sans effet au début de la dernière clause d'un prédicat
- distinction “green cut” / “red cut” : est-ce que la coupure change le comportement du programme ?

Négation

```
not(P) :- %% Cas 1: P vrai -> not(P) faux
   P, %% si P reussit
   !, %% on ignore l autre branche
   fail. %% on fait echouer le but

not(_). %% Cas 2: P faux -> not(P) vrai
```

Le dernier but est essayé uniquement quand on n'a pas réussi à prouver `P`.

Programmation logique par contraintes sur domaines finis

Problème :

- variables
- domaines des variables
- contraintes sur les variables

Schéma général

```
?- use_module(library(clpfd)).
%% utilisation bibliotheque:
%% constraint logic programming over finite domains

q([X,Y,Z]) :-
    %% domaines
    X in 100 ..1000,
    [Y, Z] ins -1000 ..199877,
    %% contraintes
    .

?- q(Vars), label(Vars).
```

- définition des domaines des valeurs
- définition des contraintes
- instanciation des variables (*labeling*)

Contraintes arithmétiques

- `A #>= B`
- `A #> B`
- `A #=< B`
- `A #< B`
- `A #= B`
- `A #\= B`

```
p1(X,Y) :-
    X > Y.
?- p1(X,Y).
```

```
p2(X,Y) :-
    [X,Y] ins -1000 ..1000,
    X #> Y.
?- p2(X,Y), label([X,Y]).
```

Contraintes réifiées

- `#\ Q` : négation
- `P #\/ Q` : ou
- `P #/\ Q` : et
- `P #\ Q` : ou exclusif
- `P #<==> Q` : équivalence
- `P #==> Q` : implication

les contraintes peuvent être manipulées comme des booléens

```
?- [X,Y] ins 0..9, X #< Y #<==> B.
```

- `B = 1` si `X #< Y`
- `B = 0` si `X #>= Y`

Bibliographie

- K.R. Apt and M. G. Wallace, 2007,
  Constraint Logic Programming using ECLiPSe,
  Cambridge University Press
- L. Sterling and E. Shapiro, 1994,
  The Art of Prolog, The MIT Press, Cambridge,
  Massachusetts, 2nd edn.