

# Visiter l'arbres de dérivation: le prédicat trace

## Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

On peut visualiser les traces des calculs en utilisant le prédicat prédéfini `trace/0` (`notrace/0` pour quitter le traceur).

Tracer un but donne lieu à une séquence de messages, qui nous informent sur l'arbre de recherche que PROLOG est en train de construire (utile pour debug).

Ces messages sont de 4 types:

- `call p(t1, ..., tn)` : le but `p(t1, ..., tn)` est en tête de la pile des buts. La recherche d'une clause dont la tête s'unifie avec `p(t1, ..., tn)` commence.
- `exit p(t1, ..., tn)`: le but `p(t1, ..., tn)` a été prouvé.
- `redo p(t1, ..., tn)`: on remonte au but `p(t1, ..., tn)` par *backtracking*, et il existe une nouvelle unification de ce but avec la tête d'une clause .
- `fail p(t1, ..., tn)`: il n'existe aucune possibilité d'unifier le but `p(t1, ..., tn)` avec la tête d'une clause.

## Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

Considérons par exemple le programme ex.pl:

$q(a).$

$q(b).$

$r(b).$

$r(c).$

$p(X) :- q(X), r(X).$

et le but:

?- P(X).

(au tableau)

# Clauses et buts disjonctifs (1)

Tracer un but

**Disjonction**

Coupure

Motivations

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

La disjonction est une abbreviation en PROLOG. La clause:

```
tete :- p_1 ; p_2 ; .... ; p_n .
```

est équivalente à la séquence de clauses:

```
tete :- p_1 . tete :- p_2 . ... tete :- p_n .
```

Donc, l'ordre des clauses d'une disjonction est important. Par exemple:

```
boucle:-boucle.
```

```
test1:- boucle>true.
```

```
test2:- true;boucle.
```

Les prédicats test1 et test2 ont la même sémantique déclarative (commutativité de la disjonction) mais le but test2. termine avec succes et le but test1. provoque une boucle.

## Clauses et buts disjonctifs (2)

Tracer un but

**Disjonction**

Coupure

Motivations

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

Analoguement, le but disjonctif

```
true;boucle
```

termine avec succes, mais

```
boucle>true
```

diverge.

Une utilisation typique de la disjonction (qui utilise le prédicat `bio` du 1er cours):

```
parent(X,Y):- bio(Y,_,_,_,X,_);bio(Y,_,_,_,_,X).
```

# Contrôler le retour en arrière

Tracer un but

Disjonction

Coupure

**Motivations**

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

- Le retour en arrière est “cablé” en Prolog.
- Il peut causer de l’inefficacité.
- On veut le contrôler.
- Pour cela, Prolog fournit un prédicat extra-logique, sans arguments: la *coupure*, ou *cut*, qui s’écrit !.

# Exemple: le maximum de deux entiers

Tracer un but

Disjonction

Coupure

**Motivations**

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

Le prédicat  $\text{max}(+Arg1, +Arg2, -Res)$ :

$\text{max}(X, Y, X) : -X \geq Y.$

$\text{max}(X, Y, Y) : -X < Y.$

Deux sources d'inefficacité:

- 1 Exclusion mutuelle non exploitée: si le premier test réussit, l'autre échoue forcément.  
Calculons l'arbre de dérivation du but  $\text{max}(2, 1, R)$ .
- 2 Exhaustivité non exploitée: si le premier test échoue, l'autre réussit forcément).  
Calculons l'arbre de dérivation du but  $\text{max}(1, 2, R)$ .

## Le maximum de deux entiers (2)

Tracer un but

Disjonction

Coupure

**Motivations**

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

C'est inefficace. Le programme C qui implemente le même algorithme serait:

```
int
max (int n,int m)
{
    if (m>=n) return m;
    if (m<n) return n;
}
```

Comment programmer un “if-then-else” en PROLOG?

## Le maximum de deux entiers (3)

Tracer un but

Disjonction

Coupure

**Motivations**

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

Ce qu'il faudrait faire: pour calculer la requête  $\max(X, Y, Z)$  on commence par évaluer  $X \geq Y$ .

Si ce but termine avec succès, on termine avec le résultat  $Z=X$  et on **coupe** l'autre branche de l'arbre de dérivation de racine  $\max(X, Y, Z)$ . Sinon, on termine avec le résultat  $Z=Y$ , **sans évaluer**  $X < Y$ .

Le prédicat prédéfini sans arguments ! permet d'obtenir ce comportement.



## Le maximum de deux entiers (4)

Tracer un but

Disjonction

Coupure

**Motivations**

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

```
/* Maximum sans coupure */
max( X, Y, X) :- X >= Y.
max( X, Y ,Y) :- X < Y.
/* Maximum avec coupure (verte):
exclusion mutuelle */
max( X, Y, X) :- X >= Y, !.
max( X, Y ,Y) :- X < Y.
/* Autre version avec coupure (rouge):
exclusion mutuelle et exhaustivite */
max( X, Y, X) :- X >= Y, !.
max( X, Y ,Y).
```

## Le maximum de deux entiers (5)

Tracer un but

Disjonction

Coupure

**Motivations**

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

$$\llbracket \text{max} \rrbracket^{eff} = \llbracket \text{max} \rrbracket^{eff} = \llbracket \text{max} \rrbracket^{eff}$$

Les trois versions de `max` ont la même sémantique *effective*.

Du point de vue de la sémantique déclarative (et opérationnelle):

$$\llbracket \text{max} \rrbracket^{decl} = \llbracket \text{max} \rrbracket^{decl}$$

en fait, on verra que du point de vue logique, `!` vaut `true`.

Mais

$$\llbracket \text{max} \rrbracket^{decl} \neq \llbracket \text{max} \rrbracket^{decl}$$

En effet, par exemple,  $\text{max}(2, 1, 1) \in \llbracket \text{max} \rrbracket^{decl} \setminus \llbracket \text{max} \rrbracket^{decl}$

# La coupure - Définition

Tracer un but

Disjonction

Coupure

Motivations

**Définition**

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

- ! est **une constante de prédicat** (c.à.d. un prédicat 0-aire, sans arguments) prédéfinie.
- Lorsqu'une instance de ! entre dans la pile des buts, au cours de la construction d'un arbre de dérivation, on appelle **but parent** de cette instance de coupure le but qui a unifié avec la tête de la clause contenant ce !.
- Lorsqu'une instance de coupure se trouve **pour la première fois** en tête de la pile des buts, elle réussit (du point de vue logique, ! est équivalent à true).
- Dès qu'un **retour en arrière** (*backtrack*) remonte à un sommets ayant un ! en première position de la pile des buts, le but parent de cette instance de coupure échoue.
- Autrement dit, tous les choix entre le "call" du but parent et le "call" de la coupure sont définitifs: tous les éventuels choix alternatifs ne seront plus considérés.

## Quelques exemples de !

Tracer un but

Disjonction

Coupure

Motivations

Définition

**Exemples**

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

```
max( X, Y, X) :- X >= Y, !.
```

```
max( X, Y ,Y).
```

```
% Membre à une seule solution
```

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

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

```
% Ajouter un élément sans duplication
```

```
ajouter( X, L, L) :- membre( X, L), !.
```

```
ajouter( X, L, [X | L]).
```

## Encore un exemple de !

Tracer un but

Disjonction

Coupure

Motivations

Définition

**Exemples**

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

$p(X) :- q(X), !, r(X).$

$p(X) :- u(X).$

$p(c).$

$q(X) :- s(X).$

$q(d).$

$r(a).$

$r(b).$

$r(d).$

$s(a).$

$s(b).$

$u(d).$

?-  $p(X).$

$X = a ;$

No

?-  $r(X), !, q(Y).$

$X = a \quad Y = a ;$

$X = a \quad Y = b ;$

$X = a \quad Y = d ;$

No

## Le if then else

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

Avec la coupure, on peut définir le `if then else`. Il s'agit d'une macro prédéfinie en Prolog.

`if P then Q else R`  
s'écrit

`P -> Q ; R.`

et est défini par:

`P -> Q ; R :- P, !, Q.`

`P -> Q ; R :- R.`

Exemple d'utilisation:

`max(X,Y,Z) :- X>= Y -> Z=X ; Z=Y.`

`boucle :- write("Entrez un entier: "),  
read(N),  
(N==0 -> true; boucle)`

## La négation

Prolog fournit un prédicat unaire qui implemente une forme de négation. Ce prédicat s'écrit `not`.

- Le but `not(B)` réussit si l'arbre de dérivation du but `B` est fini et n'a aucune feuille succès.

- Ce n'est pas la négation logique:

Par exemple, pour le programme `P`:

`max(X,Y,Z) :- X>=Y, !, Z=Y.`

`max(_,Y,Y).`

le but:

`not(max(2,1,1)).`

réussit, même si  $\max(2,1,1) \in \llbracket P \rrbracket^{decl}$ .

Par exemple, pour le programme:

`p(X) :- ¬p(X).`

le but:

`P(X);not(P(X)).`

ne réussit pas (même si  $p(X) \vee \neg p(X)$  est une formule valide).

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

**Négation**

Prédicats

d'entrée-sortie

Prédicats du

second ordre

Résumé

## La négation (2)

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

**Négation**

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

`not(B)` est défini par:

```
not(B) :- B, !, fail.
```

```
not(B).
```

Attention: nier des prédicats qui contiennent des variables non instanciées peut donner des résultats inattendus, comme dans l'exemple suivant:



## La négation (3)

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

**Négation**

Prédicats

d'entrée-sortie

Prédicats du

second ordre

Résumé

```
marie(francois).
```

```
etudiant(rene).
```

```
etudiant_celibataire(X) :-
```

```
    not(marie(X)),
```

```
    etudiant(X).
```

```
?- etudiant_celibataire(X).
```

```
no
```

```
?- etudiant_celibataire(rene).
```

```
yes
```

## La négation (4)

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

**Négation**

Prédicats

d'entrée-sortie

Prédicats du

second ordre

Résumé

La “bonne version” du programme

```
marie(francois).
```

```
etudiant(rene).
```

```
etudiant_celibataire(X) :- etudiant(X), not(marie(X)).
```

Il faut que l'appel d'un but avec négation soit fait *après* l'instanciation des variables.

# Entrée-Sortie

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

**Prédicats  
d'entrée-sortie**

Prédicats du  
second ordre

Résumé

- ECLiPSe communique avec des flux (streams).
- Il y a un flux d'entrée standard `stdin` et de sortie standard `stdout`. Il est possible de définir des streams pour faire de l'entré-sortie sur des fichiers.

Les prédicats de base:

- `open(FileName, Mode, S)`. ouvre un fichier en lecture (`Mode=read`) ou écriture (`Mode=write`). `S` est le stream correspondant.
- `read(S, X)` lit le terme `X` du stream `S`.
- `write(S, X)` écrit le terme `X` sur le stream `S`.
- `close(S)` ferme le flux `S`.

## Exemple 1

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

**Prédicats  
d'entrée-sortie**

Prédicats du  
second ordre

Résumé

```
cube :-
write(" Entrez un nombre (ou stop): "),
/* equivaut à write(stdout, ..) */
read(X),
/* equivaut à read(stdin, X) */
process(X).

process(stop).

process(N) :- C is N * N * N, write("Le cube de "),
write(N),
write(" est: "),
write(C), nl, cube.
```

## Exemple 2

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

**Prédicats  
d'entrée-sortie**

Prédicats du  
second ordre

Résumé

```
/* lire un par un les termes du fichiers File,  
et les copier dans la liste Result */
```

```
read_data(File,Result):-  
    open(File,read,S),  
    read(S,X),  
    read_data_lp(S,X,Result),  
    close(S).
```

```
read_data_lp(S,end_of_file,[]):- !.
```

```
read_data_lp(S,X,[X|R]):-  
    read(S,Y),  
    read_data_lp(S,Y,R).
```

# Prédicats du second ordre

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

- `bagof(X, P, L)` produit la liste L de tous les termes t tel que le but  $P[t/X]$  est satisfait.
- `setof(X, P, L)` comme `bagof` sauf que la liste L est ordonné et les doublons sont éliminés.

```
age(marie,5).
```

```
age(anne,5).
```

```
age(paul,7).
```

```
age(marc,10).
```

```
[eclipse 2]: bagof(Enfant,age(Enfant,5),Liste).
```

```
Enfant = Enfant
```

```
Liste = [marie, anne]
```

```
Yes (0.00s cpu)
```

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

```
[eclipse 3]: bagof(Enfant,age(Enfant,Age),Liste).
```

```
Enfant = Enfant
```

```
Age = 5
```

```
Liste = [marie, anne]
```

```
Yes (0.00s cpu, solution 1, maybe more) ? ;
```

```
Enfant = Enfant
```

```
Age = 7
```

```
Liste = [paul]
```

```
Yes (0.00s cpu, solution 2, maybe more) ? ;
```

```
Enfant = Enfant
```

```
Age = 10
```

```
Liste = [marc]
```

```
Yes (0.00s cpu, solution 3)
```

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

Résumé

```
[eclipse 4]: setof(Enfant,age(Enfant,Age),Liste).
```

```
Enfant = Enfant
```

```
Age = 5
```

```
Liste = [anne, marie]
```

```
Yes (0.00s cpu, solution 1, maybe more) ? ;
```

```
Enfant = Enfant
```

```
Age = 7
```

```
Liste = [paul]
```

```
Yes (0.00s cpu, solution 2, maybe more) ? ;
```

```
Enfant = Enfant
```

```
Age = 10
```

```
Liste = [marc]
```

```
Yes (0.00s cpu, solution 3)
```



# Résumé du troisième cours

Tracer un but

Disjonction

Coupure

Motivations

Définition

Exemples

If then else

Négation

Prédicats  
d'entrée-sortie

Prédicats du  
second ordre

**Résumé**

- Tracer un but.
- La disjonction.
  - La coupure.
  - Applications de !: conditionnelles et négation.
- Prédicats d'entrée/sortie.
- Prédicats du second ordre.