Programmation logique
PROLOG

D.E ZEGOUR
École Supérieure d'Informatique
ESI

Programmation logique / PROLOG

Sommaire

Programmation logique / PROLOG

Concepts généraux

PROLOG = PROgrammation LOGique

Créé vers les années 1970.

interrogation de base de données

Conception assistée par ordinateur

Compréhension du langage naturel

PROLOG

Réalisation de systèmes experts

Base de données déductives

Programmation logique / PROLOG

Concepts généraux

Particularité :

- Aucune distinction entre programme et données,
- Aucune structure de contrôle.

Programmer en logique consiste à

- définir les hypothèses
- énoncer la conclusion

Prolog implémente la règle de *résolution descendante utilisant la technique du Backtracking*
(recherche en profondeur et retour arrière)

Exécuter un programme = demander la preuve

Propriété de l' "*Invertibility*" : possibiliter de trouver tous les objets en relation avec d'autres objets.

Programmation logique / PROLOG

Données manipulées

Les types

standard : integer, real, char, symbol

construit (voir les structures)

Objets élémentaires : variables et constantes.

- Les variables sont des combinaisons de lettres, chiffres et _. Le premier caractère est une lettre alphabétique majuscule ou '_'.
- Les constantes peuvent être des nombres, symbole(chaîne de caractères dont la première lettre est en minuscule, ou une chaîne de caractères quelconque entre ".

Programmation logique / PROLOG

Données manipulées

Listes

Définition d'une liste L de type

type : L = type*

dans la partie Domains

[ ] désigne une liste vide.

[a, b, c] liste formée des 3 éléments a, b et c.

Dans l'écriture [X|Y]

X désigne le premier élément de la liste

Y la liste sans le premier élément

[a, b, c] = [a|[b, c]] = [a, b|[c]] = [a, b, c|[ ]]

Exemples

Dans la partie Domains

Liste : ensemble d'entiers

Liste = integer*

Matrice : ensemble de listes

Matrice = Liste*

Objets composés

typedate = date(integer, integer, integer)

Programmation logique / PROLOG

Structure d'un programme

Domains

Construction de nouveaux types

Predicates

Définitions des prédicats(types des objets en relation)

Clauses

Faits et règles

Goal

But

Les constantes ont une portée globale.

La portée d'une variable se limite à une clause.

:- ou if

, veut dire et logique

; veut dire ou logique

Programmation logique / PROLOG

Structure interne

La objets manipulés sont représentés en arbres.

Prolog se comporte comme un démonstrateur de théorème.

L'interpréteur Prolog possède donc son propre mécanisme de résolution.

Il utilise les deux principes suivants : Résolution et Unification

Autres opérations

! : Arrêt du processus du backtracking dés qu'une solution est trouvée

Les règles peuvent être récursives.

Autres opérations :

- Communication avec l'extérieur
- Calcul arithmétique et logique
- Saisie et mise à jour des règles.

Exemples de programmes Prolog
(Programme 1)

```
Predicates
        Parent(symbol, symbol)
        Frere(symbol, symbol)
        Ascendant(symbol, symbol)
Clauses
        Parent(aa, bb).
        Parent(bb, cc) .
        Parent(bb, dd).
        Parent(dd, ee).
        Parent(ff, dd).

Frere(X, Y) :- Parent(Z, X), Parent(Z, Y), Not(X=Y).
Ascendant(X,Y)  :- Parent(X, Y).
Ascendant(X,Y)  :- Parent(X, Z), Ascendant(Z,Y).

/* On peut aussi écrire ascendant comme suit :

Ascendant(X,Y)  :- Parent(X, Y);
Parent(X, Z),Ascendant(Z, Y).

*/
```

Programmation logique / PROLOG

Parent(aa, bb).
Parent(bb, cc) .
Parent(bb, dd).
Parent(dd, ee).
Parent(ff, dd).

Question : Frere(dd, cc) Réponse Yes

Question : Frere(X, dd) Réponse X = cc

Question : Frere(dd, X) Réponse X = cc

Question : Ascendant(X, ee)

Réponse

X = dd

X = aa

X = bb

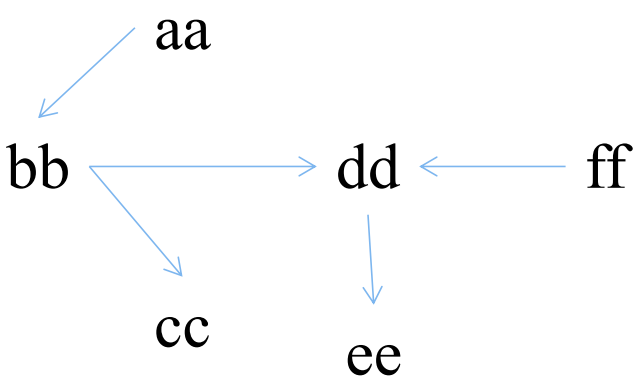
X = ff

Question : Ascendant(X, cc)

Réponse

X = bb

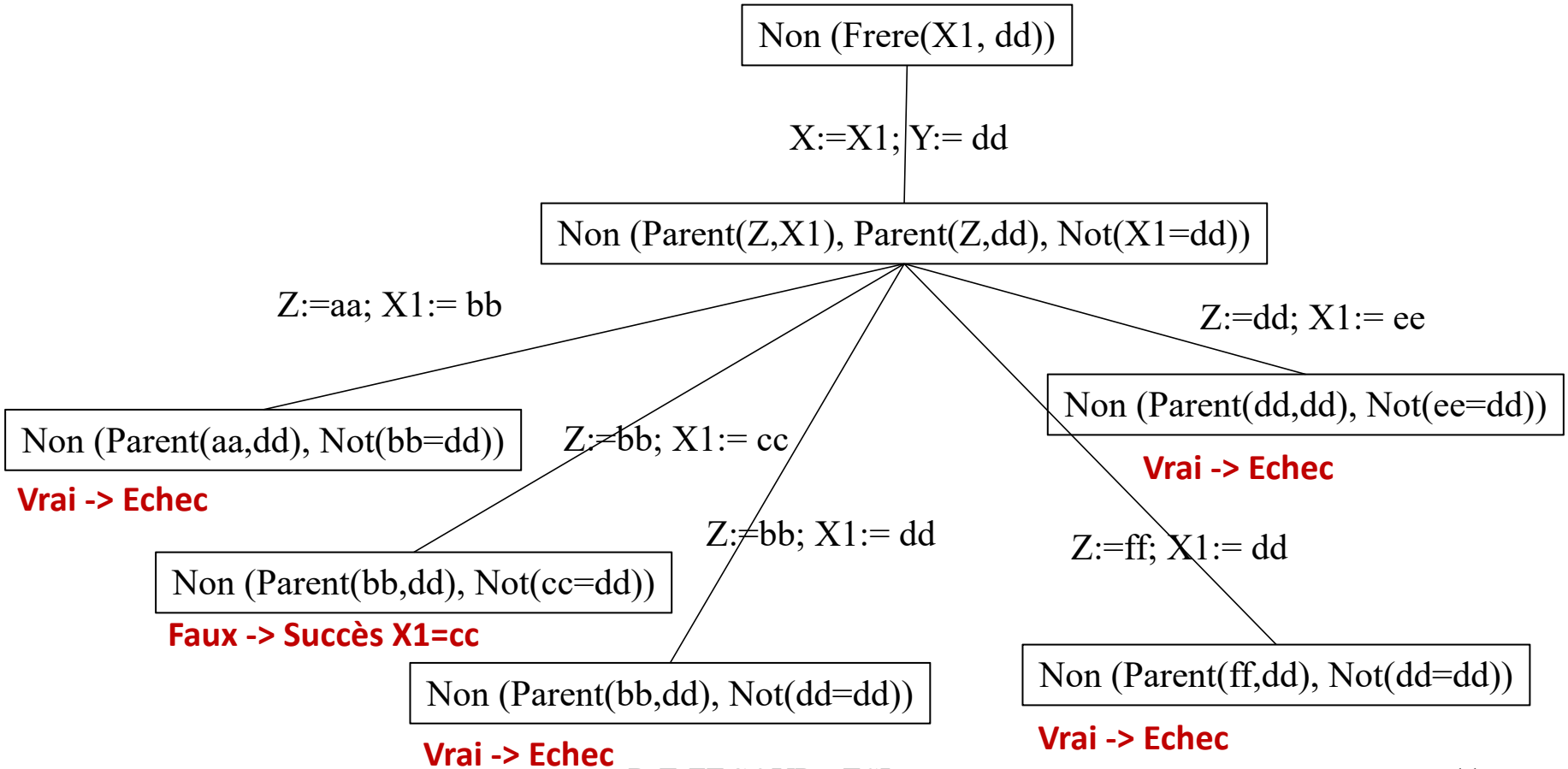
X = aa



Programmation logique / PROLOG

Parent(aa, bb).
Parent(bb, cc) .
Parent(bb, dd).
Parent(dd, ee).
Parent(ff, dd).
Frere(X, Y) :- Parent(Z, X), Parent(Z, Y), Not(X=Y).

Fonctionnement de Prolog

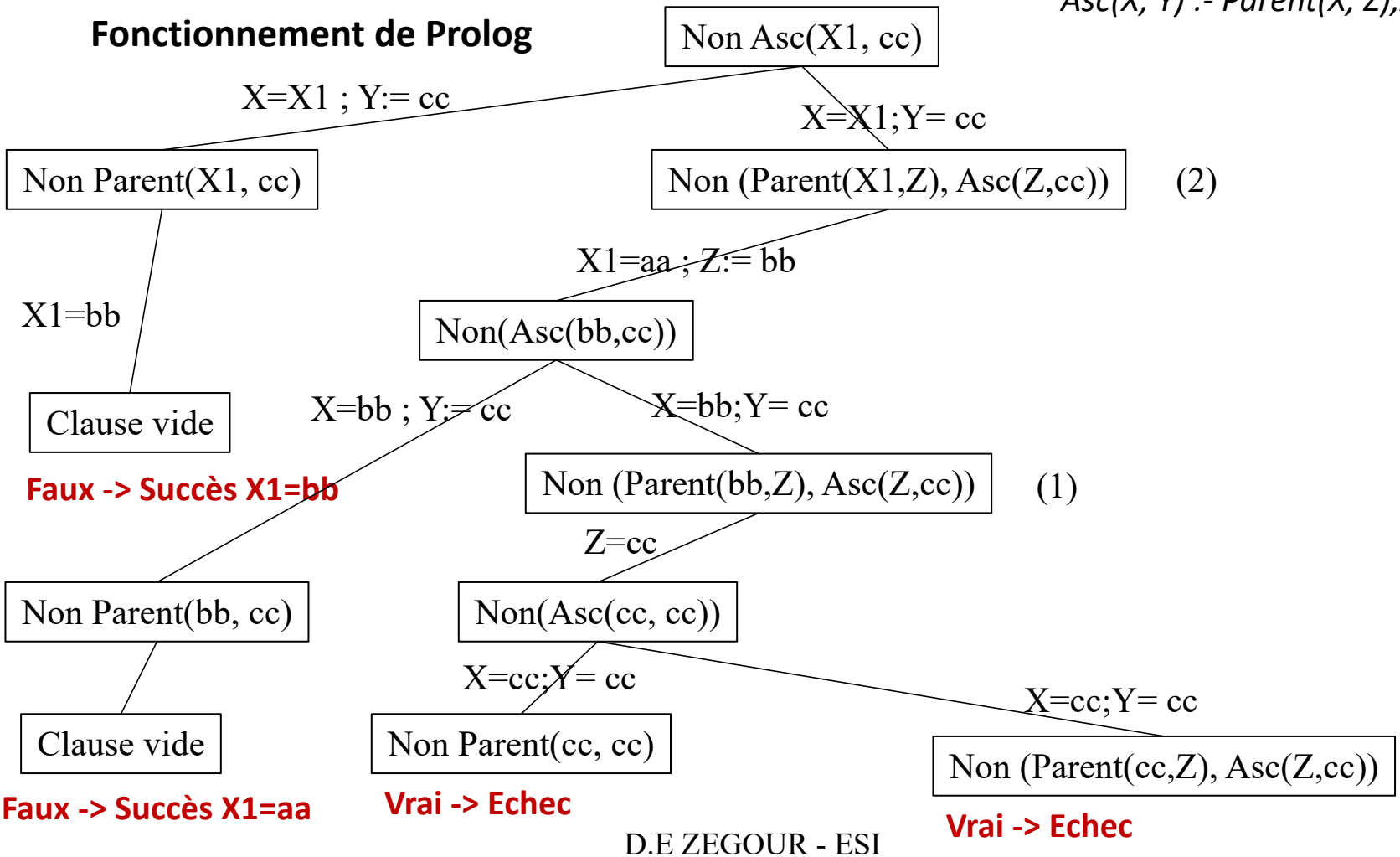


Question : Frere(X, dd)

Programmation logique / PROLOG

Parent(aa, bb).
Parent(bb, cc) .
Parent(bb, dd).
Parent(dd, ee).
Parent(ff, dd).
Asc(X, Y) :- Parent(X, Y).
Asc(X, Y) :- Parent(X, Z),Asc(Z,Y).

Fonctionnement de Prolog

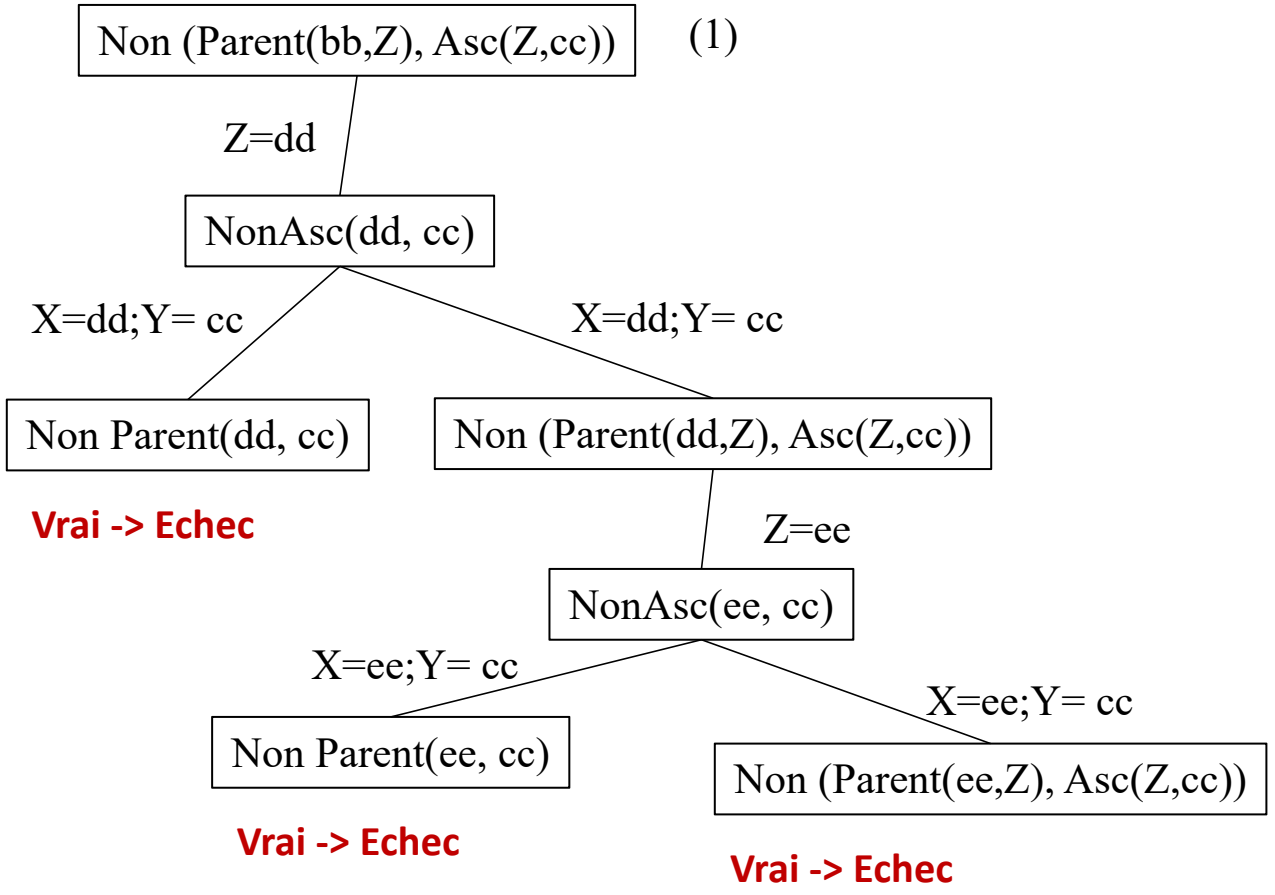


Question : Asc (X, cc)

Programmation logique / PROLOG

Parent(aa, bb).
Parent(bb, cc) .
Parent(bb, dd).
Parent(dd, ee).
Parent(ff, dd).
Asc(X, Y) :- Parent(X, Y).
Asc(X, Y) :- Parent(X, Z),Asc(Z,Y).

Fonctionnement de Prolog

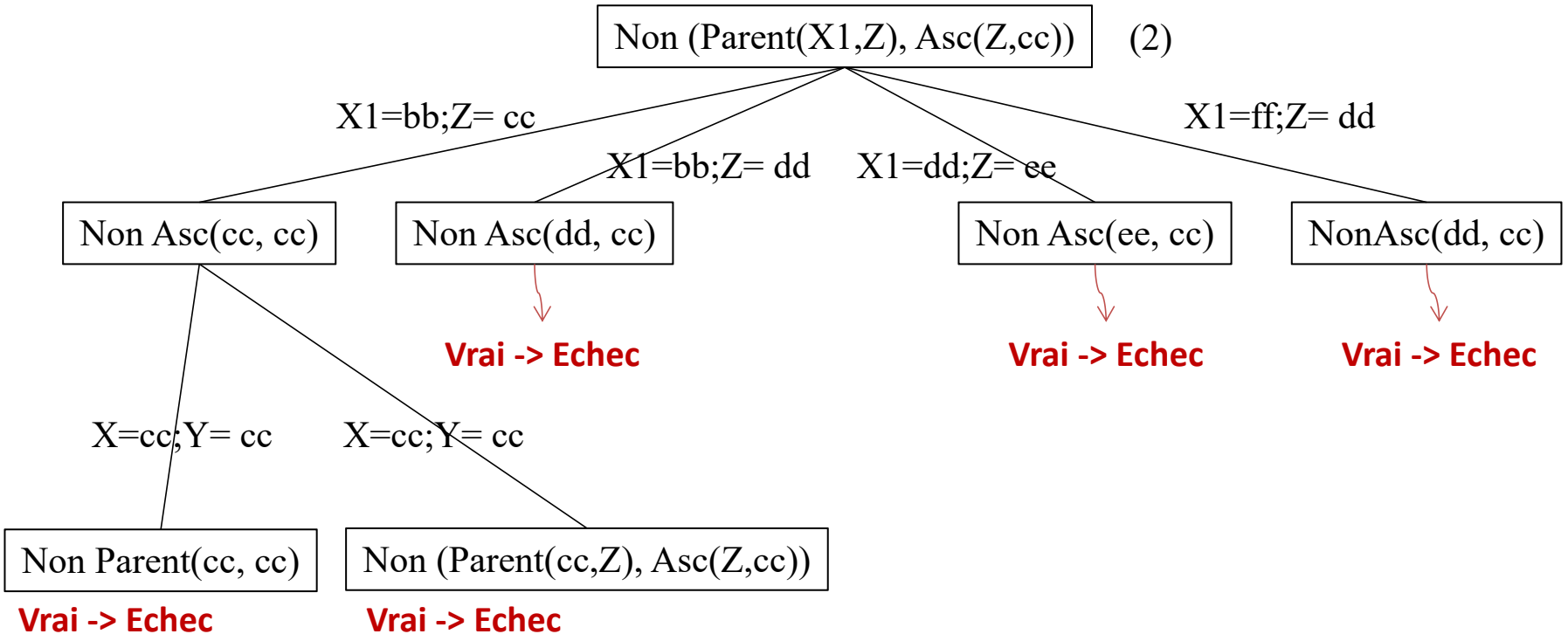


Question : Asc (X, cc)

Programmation logique / PROLOG

Parent(aa, bb).
Parent(bb, cc) .
Parent(bb, dd).
Parent(dd, ee).
Parent(ff, dd).
Asc(X, Y) :- Parent(X, Y).
Asc(X, Y) :- Parent(X, Z),Asc(Z,Y).

Fonctionnement de Prolog



Question : Asc (X, cc)

Programmation logique / PROLOG

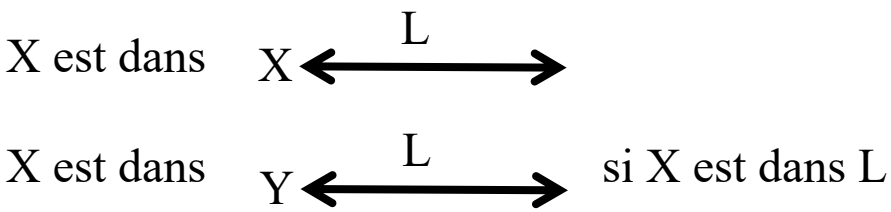
Exemples de programmes Prolog (Programme 2)

Appartenance d'un élément à une liste:

App(X, [X|L]).
App(X, [Y|L]):-App(X,L)

Modélisation

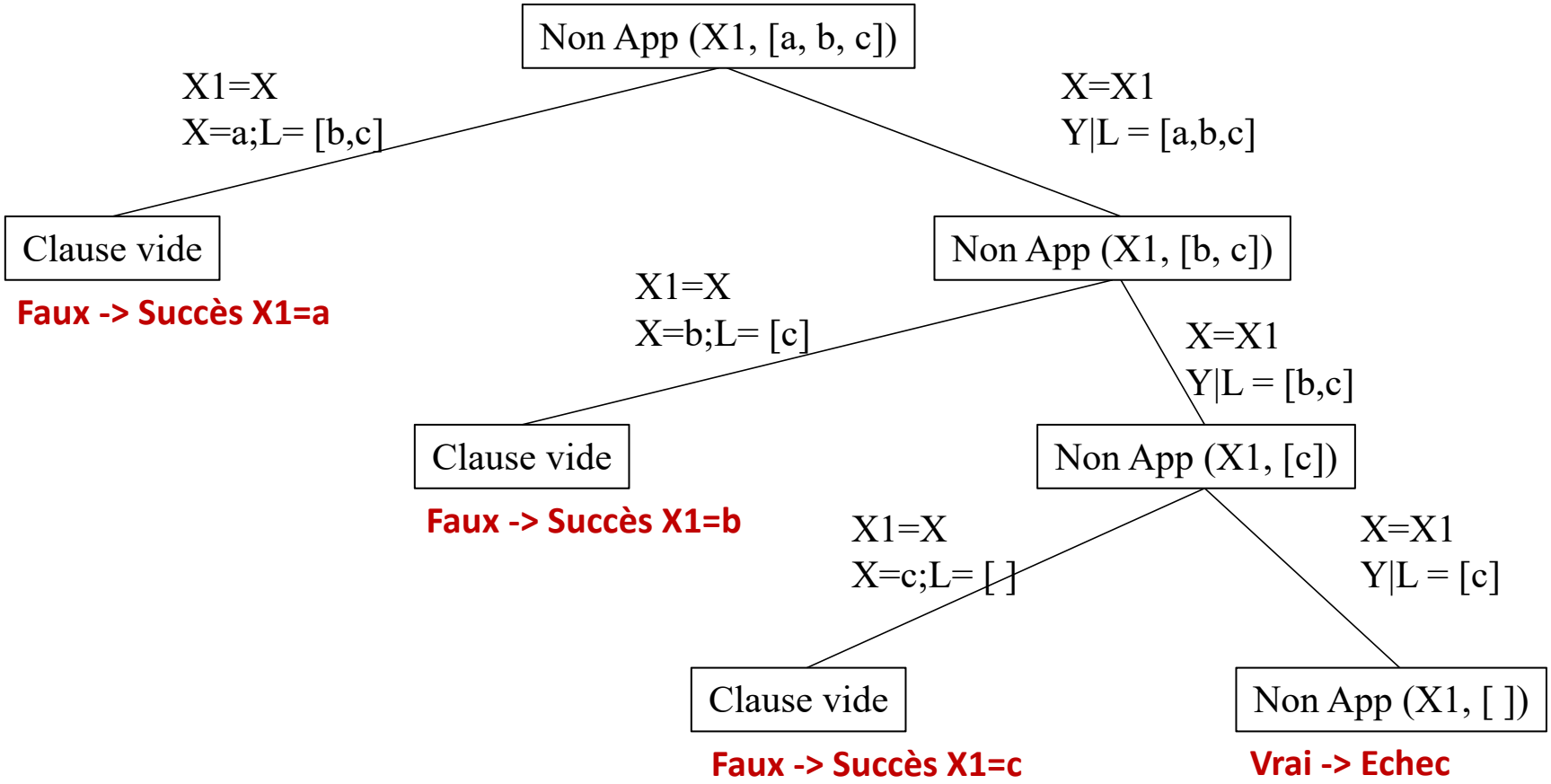
App (X, L) : Vrai si X appartient à la liste L



App(X, [X|L]).
App(X, [Y|L]):-App(X,L)

Programmation logique / PROLOG

Fonctionnement de Prolog



Question : App(X, [a, b, c])

Programmation logique / PROLOG

Exemples de programmes Prolog (Programme 3)

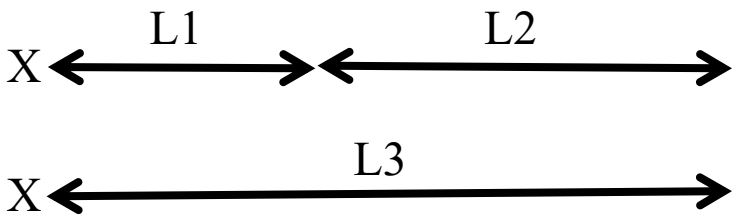
Concaténation de 2 listes

Cat([ ], L, L).

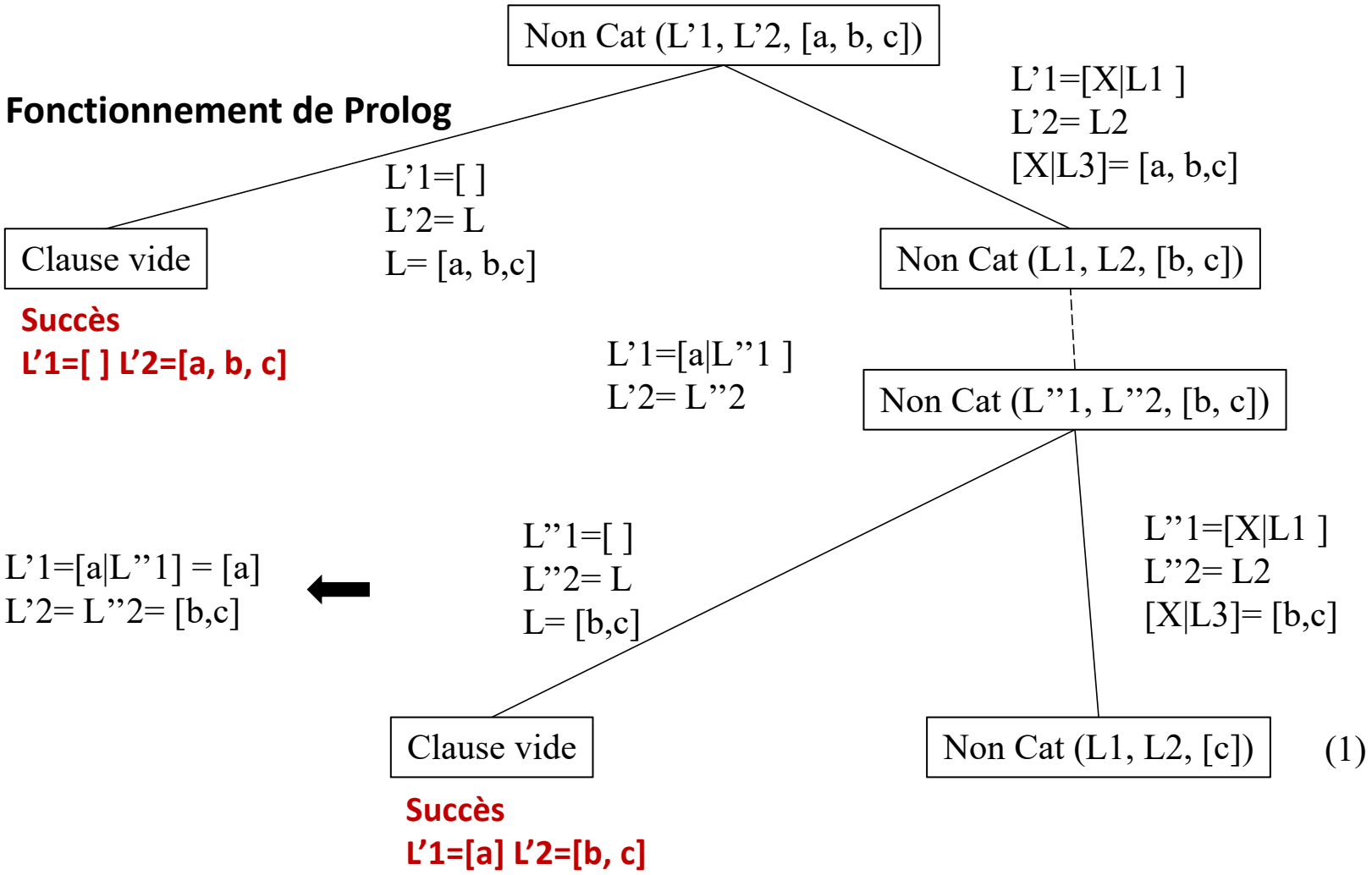
Cat ([X|L1], L2, [X|L3]):-Cat(L1, L2, L3)

Modélisation

Cat (L1, L2, L3) : Vrai si L3 est la concaténation de L1 et L2



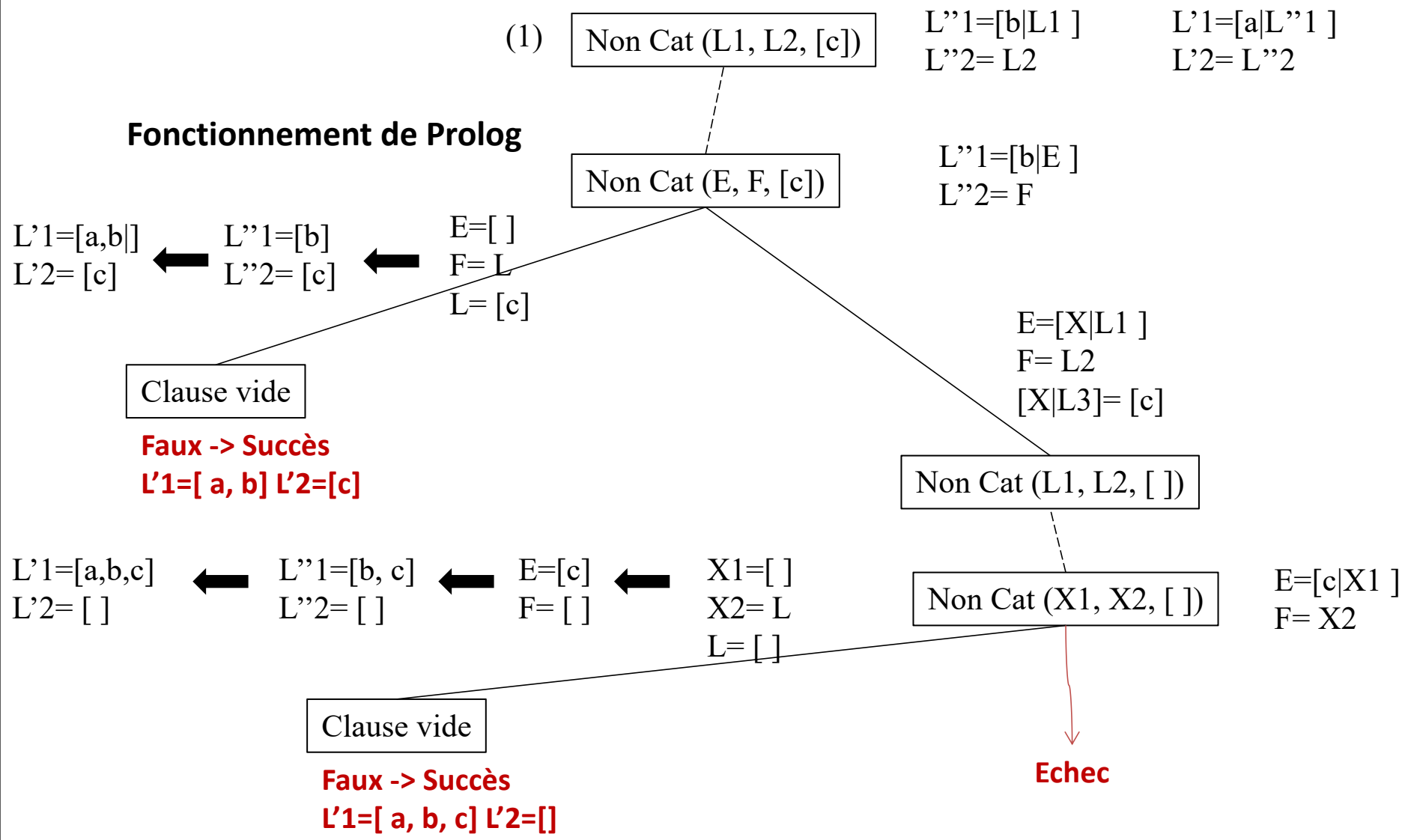
Programmation logique / PROLOG



Question : Cat(L1, L2, [a, b, c])

Programmation logique / PROLOG

Cat([ ], L, L).
Cat([X|L1], L2, [X|L3]):-Cat(L1, L2, L3)



Question : Cat(L1, L2, [a, b, c])

Exemples de programmes Prolog (Programme 4)

Supprimer un élément d'une liste

Sup([X|L], X, L).

Sup([Y|L1], X, [Y|L2]):-Sup(L1, X, L2)

Modélisation

Sup (L1, X, L2) : Vrai si L2 est L1 sans l'élément X

Supprimer X de X ⟷ (L) C'est L

Supprimer X de Y ⟷ (L1)

C'est Y ⟷ (L2) Si L2 est L1 sans l'élément X

Exemples de programmes Prolog (Programme 5)

Eclater une liste L en deux sous listes L1 et L2

L1 : éléments de rang impair

L2 : éléments de rang pair

Eclater([ ], [ ], [ ]).

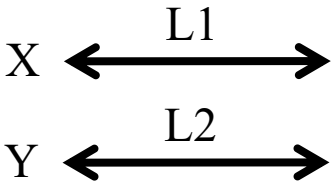
Eclater([X], [X], [ ]).

Eclater([X|[Y|L]], [X|L1], [Y|L2]):-
Eclater(L, L1, L2)

Modélisation

Eclater (L, L1, L2) : Vrai si L1 contient les éléments de L de rang pair et L2 contient les éléments de L de rang impair.

Donne



Si Eclater (L, L1, L2)

Exemples de programmes Prolog (Programme 6)

Longueur d'une liste

```
Long([ ], 0).
Long[[ _ |L], N) :- N1=N-1,  Long[L, N1)
```

Exemples de programmes Prolog (Programme 7)

Affichage d'une liste

```
Afficher([ ]).
Afficher([X|L]) :- write(X), nl, Afficher(L)
```

Programmation logique / PROLOG

Exemples de programmes Prolog (Programme 8)

Que fait le programme suivant ?

```
Cube :- write(''un nombre : '', readln(X), Traiter(X).
Traiter(stop) :- !
Traiter(N) :- C = N*N*N,
              write(''Le cube de ''), write(N), write('' est ''),
              write(C), nl, Cube.
```

--> Affichage des cubes d'une suite de nombres lus un à un terminée par stop.