Paradigme de programare

Laborator 10

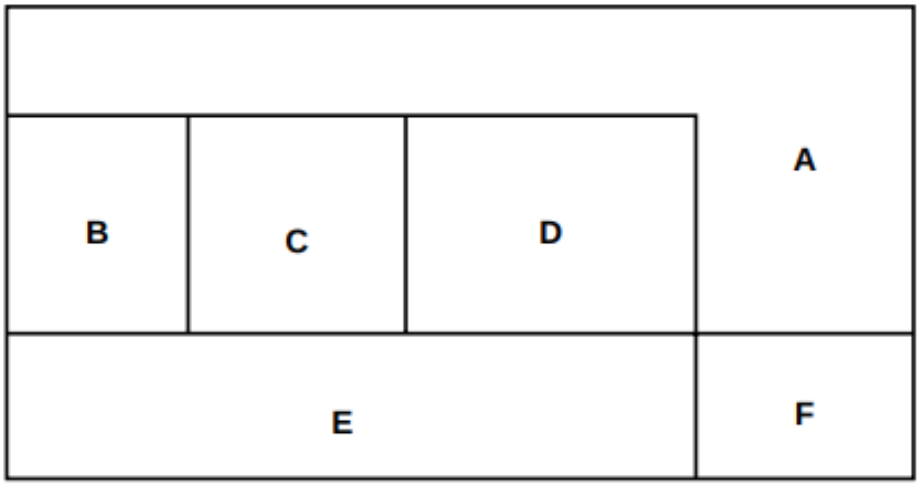
Mihai Nan

Facultatea de Automatica si Calculatoare
Universitatea Politehnica din Bucuresti

Anul universitar 2020-2021

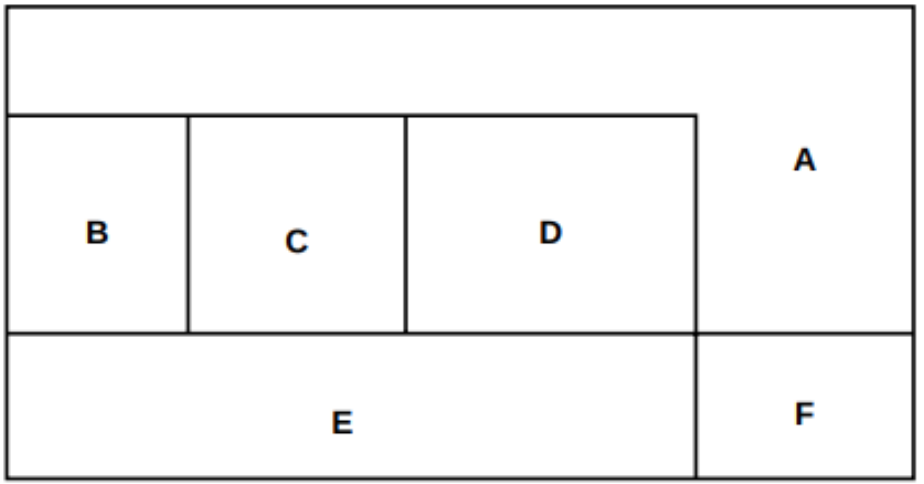
Problema colorării hărților

- Pornind de la următoarea hartă, dorim să colorăm fiecare celulă cu o culoare astfel încât să nu existe două celule adiacente colorate cu aceeași culoare.



Problema colorării hărților

- Pornind de la următoarea hartă, dorim să colorăm fiecare celulă cu o culoare astfel încât să nu existe două celule adiacente colorate cu aceeași culoare.




```
coloring(A,B,C,D,E,F) :- different(A,B), different(A,C),
  different(A,D), different(A,F), different(B,C),
  different(B,E), different(C,D), different(C,E),
  different(D,E), different(E,F).
```

Problema colorării hărților

```
coloring(A,B,C,D,E,F) :- different(A,B), different(A,C),
  different(A,D), different(A,F), different(B,C),
  different(B,E), different(C,D), different(C,E),
  different(D,E), different(E,F).

different(yellow, blue). different(yellow, red).  different(blue, red).
different(blue, yellow). different(red, yellow).  different(red, blue).
```

Interogare: `?- coloring(A,B,C,D,E,F).`

A = yellow, B = blue, C = ~~blue~~, D = ~~blue~~, F = blue

C = red, D = blue, F = blue, E = yellow

Problema colorării hărților

```
coloring(A,B,C,D,E,F) :- different(A,B), different(A,C),
  different(A,D), different(A,F), different(B,C),
  different(B,E), different(C,D), different(C,E),
  different(D,E), different(E,F).

different(yellow, blue). different(yellow, red).  different(blue, red).
different(blue, yellow). different(red, yellow).  different(red, blue).
```

Interogare: `?- coloring(A,B,C,D,E,F).`

```
A = E, E = yellow,
B = D, D = F, F = blue,
C = red ;
```

Problema colorării hărților

```
coloring(A,B,C,D,E,F) :- different(A,B), different(A,C),
  different(A,D), different(A,F), different(B,C),
  different(B,E), different(C,D), different(C,E),
  different(D,E), different(E,F).

different(yellow, blue). different(yellow, red).  different(blue, red).
different(blue, yellow). different(red, yellow).  different(red, blue).
```

Interogare: `?- coloring(A,B,C,D,E,F).`

```
A = E, E = yellow,
B = D, D = F, F = blue,
C = red
```

```
A = E, E = yellow,
B = D, D = blue,
C = F, F = red ;
```

Problema colorării hărților

```
coloring(A,B,C,D,E,F) :- different(A,B), different(A,C),
  different(A,D), different(A,F), different(B,C),
  different(B,E), different(C,D), different(C,E),
  different(D,E), different(E,F).

different(yellow, blue). different(yellow, red).  different(blue, red).
different(blue, yellow). different(red, yellow).  different(red, blue).
```

Interogare: `?- coloring(A,B,C,D,E,F).`

```
A = E, E = yellow,
B = D, D = F, F = blue,
C = red

A = E, E = yellow,
B = D, D = blue,
C = F, F = red

A = E, E = blue,
B = D, D = F, F = yellow,
C = red ;
```

...

Problema colorării hărților

```
different(yellow, blue). different(yellow, red).  different(blue, red).
different(blue, yellow). different(red, yellow).  different(red, blue).
```

- Cum putea păstra doar prima soluție?

Problema colorării hărților

```
different(yellow, blue). different(yellow, red).  different(blue, red).
different(blue, yellow). different(red, yellow).  different(red, blue).
```

- Cum putea păstra doar prima soluție?
- **Soluție:** Oprim recursivitatea după prima satisfacere a scopului (folosind operatorul `cut`)!

```
coloring(A,B,C,D,E,F) :- different(A,B), different(A,C),
  different(A,D), different(A,F), different(B,C),
  different(B,E), different(C,D), different(C,E),
  different(D,E), different(E,F), !.
```

Rolul lui cut

Oprirea backtracking-ului la prima satisfacere a unui anumit scop.
În sensul că se vor încerca în continuare toate soluțiile care se pot obține din acest punct în dreapta, dar nu vom încerca să resatisfacem vreun scop din trecut

Aplicație pentru ! (cut)

```
dacă p(X) atunci
        q(X)
altfel
        r(X)
```

Soluție:

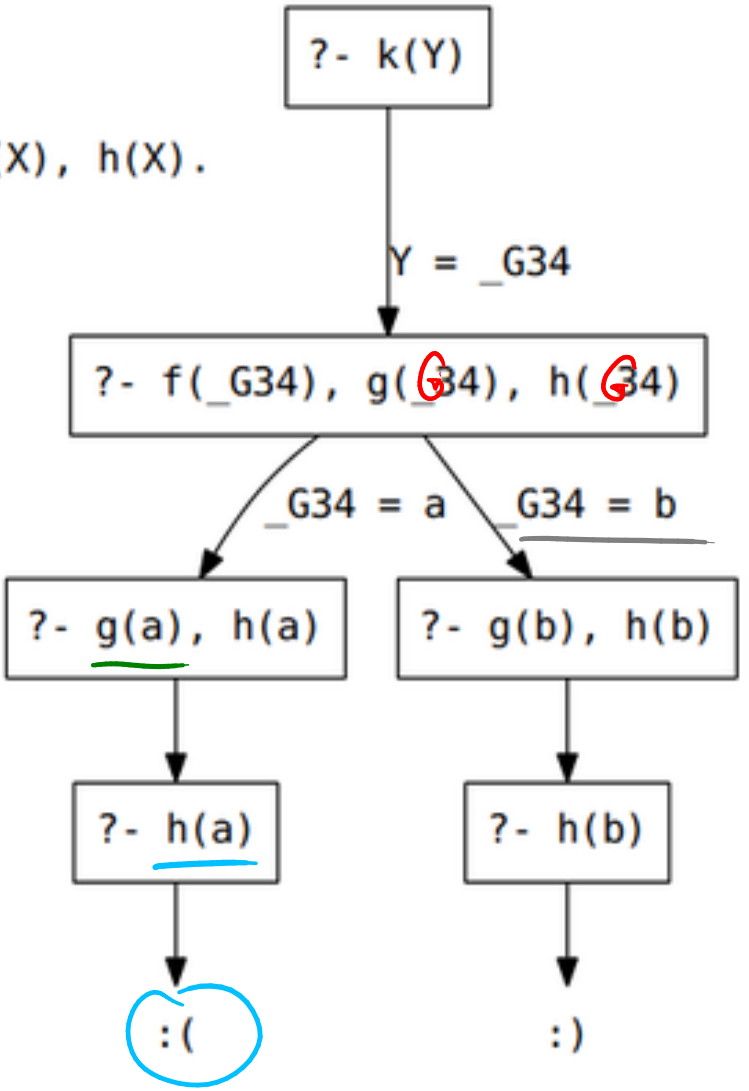
```
pred(X) :- p(X), !, q(X).
pred(X) :- r(X).
```

Exemplu: Calculul factorialului

```
factorial(N,F) :- N =< 0, !, F is 1.
factorial(N, F) :- N1 is N - 1, factorial(N1, R), F is R * N.
```

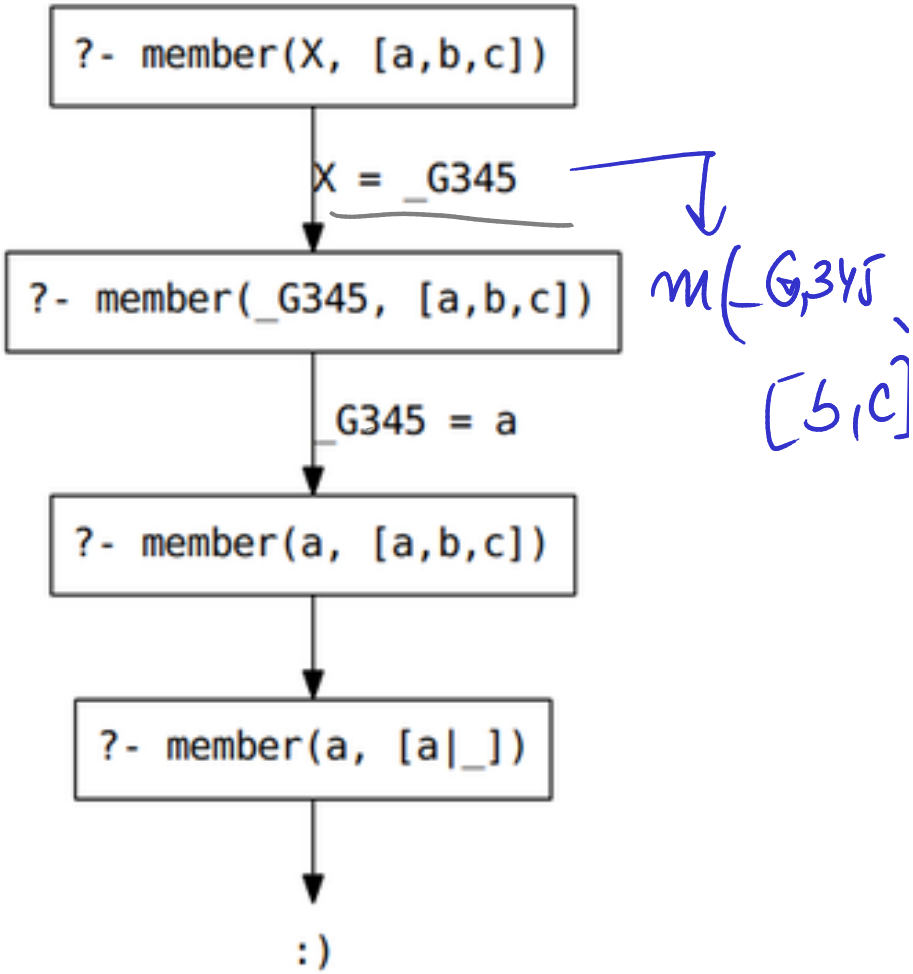
Exemplul I

```
f(a).
f(b).
g(a).
g(b).
h(b).
k(X) :- f(X), g(X), h(X).
```



Exemplul II

```
member(X, [X|_]).
member(X, [_|T]) :- member(X, T).
```



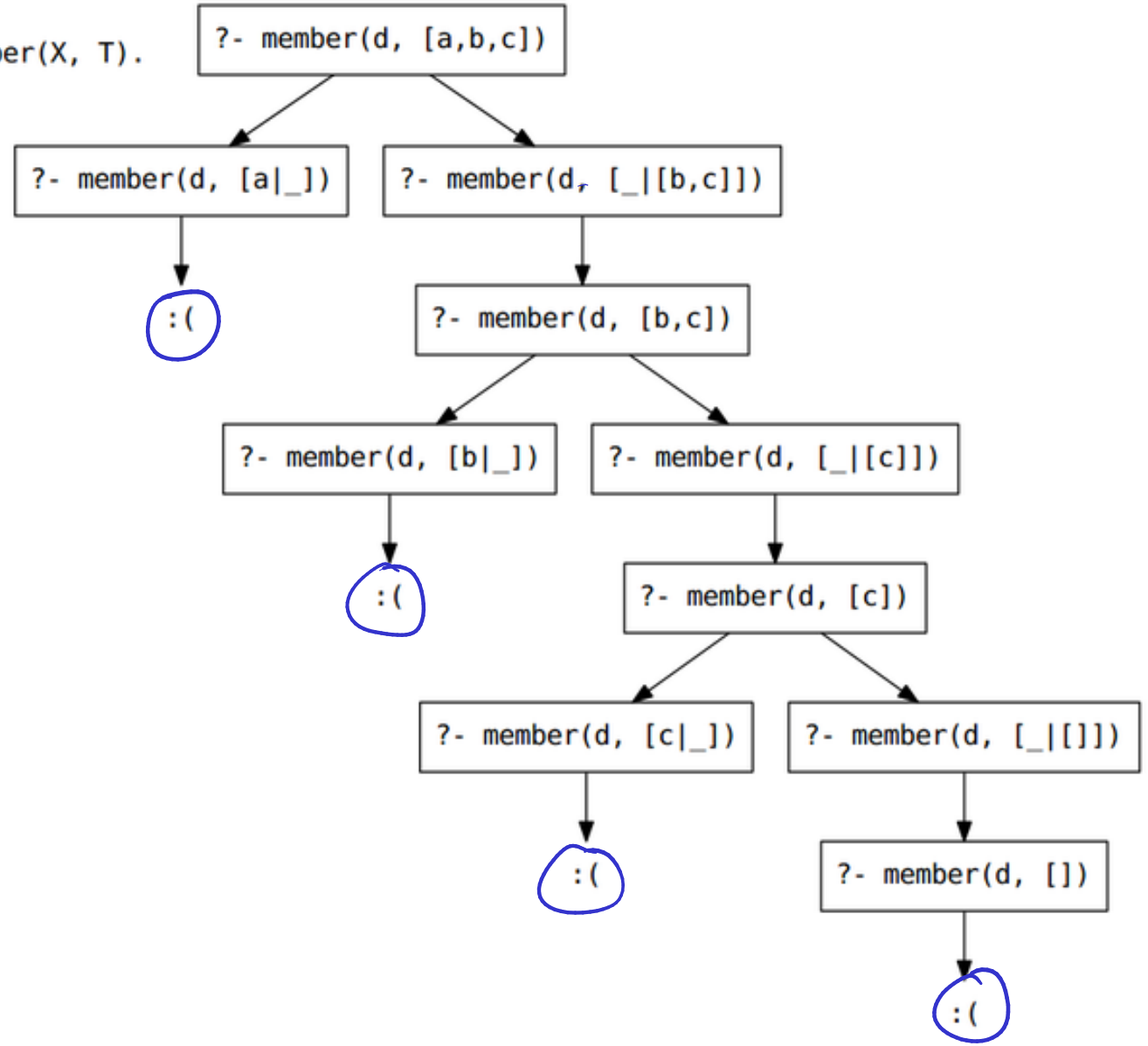
X = a ;

X = b ;

X = c .

Exemplul III

```
member(X, [X|_]).
member(X, [_|T]) :- member(X, T).
```



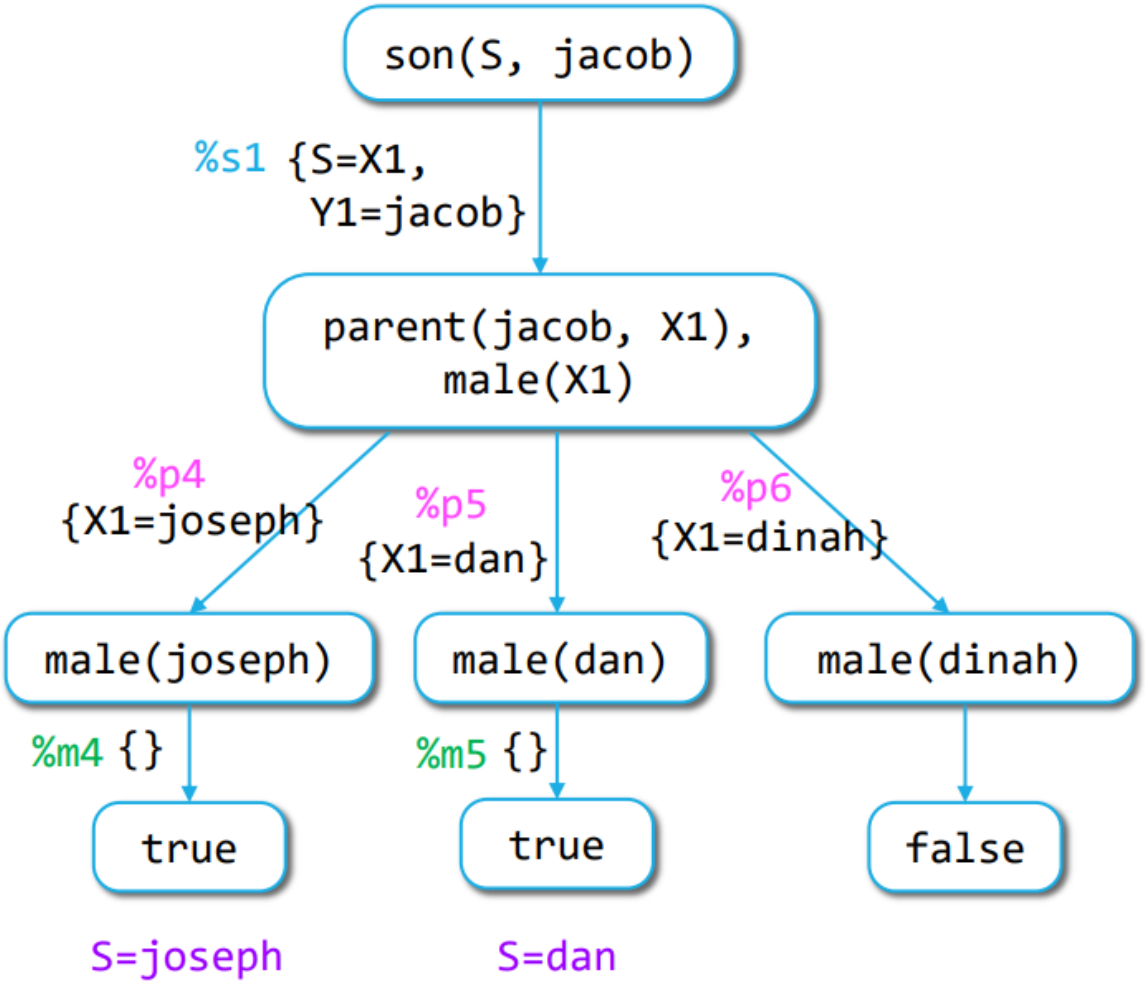
Exemplul IV

```
parent(abraham, isaac). %p1
parent(isaac, jacob).   %p2
parent(sarah, isaac).   %p3
parent(jacob, joseph).  %p4
parent(jacob, dan).     %p5
parent(jacob, dinah).   %p6

male(abraham).          %m1
male(isaac).            %m2
male(jacob).            %m3
male(joseph).           %m4
male(dan).              %m5

son(X,Y) :- parent(Y,X), %s1
            male(X).
```

```
?- son(S,jacob).
```

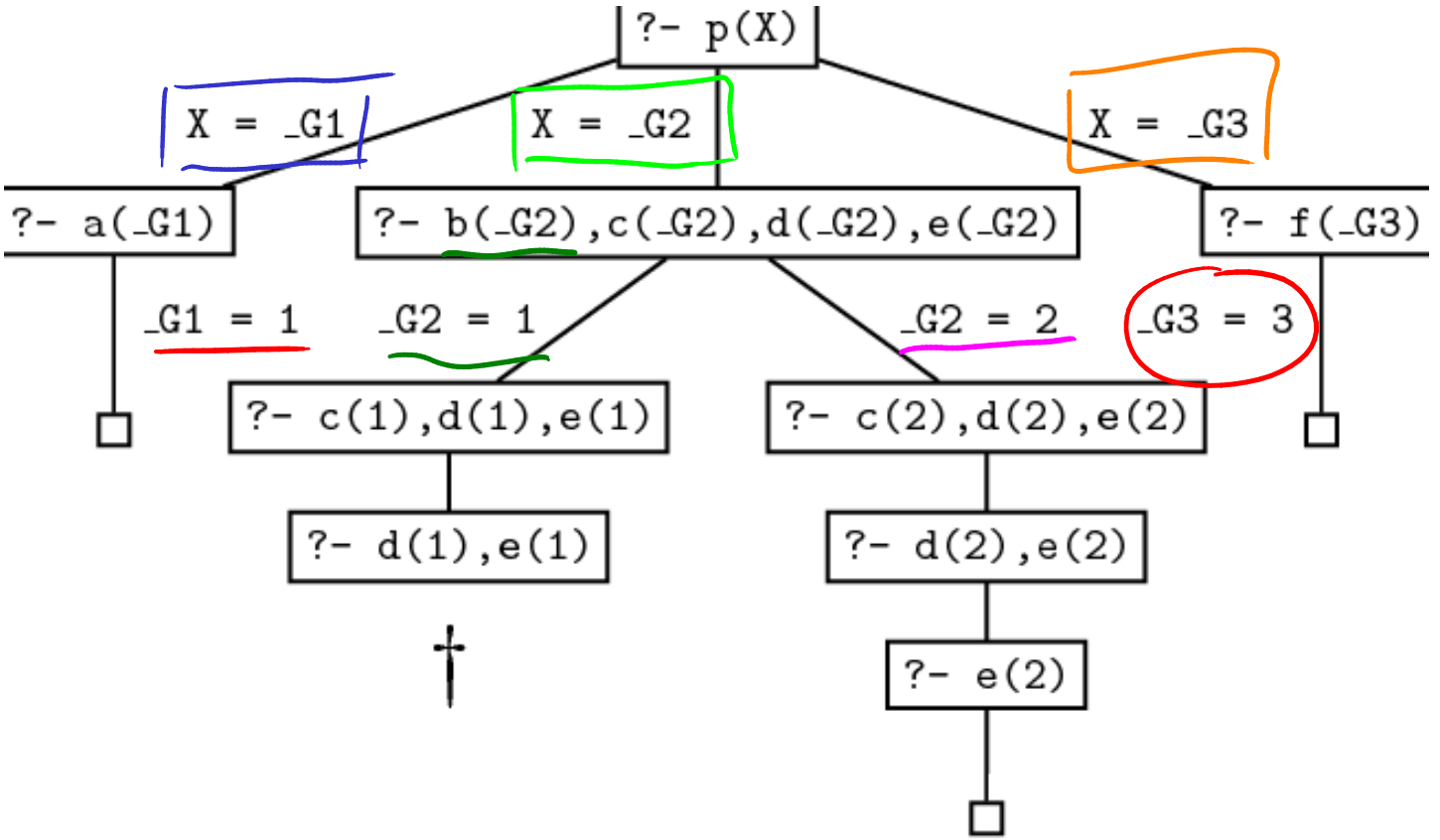


Exemplul V

```
p(X):- a(X).
p(X):- b(X),c(X),d(X),e(X).
p(X):- f(X).
a(1).
```

```
b(1).
b(2).
c(1).
c(2).
```

```
d(2).
e(2).
f(3).
```



Exemplul VI (cu !)

```
p(X):- a(X).
p(X):- b(X),c(X),!,d(X),e(X).
p(X):- f(X).
a(1).
```

```
b(1).
b(2).
c(1).
c(2).
```

```
d(2).
e(2).
f(3).
```

Explicații:

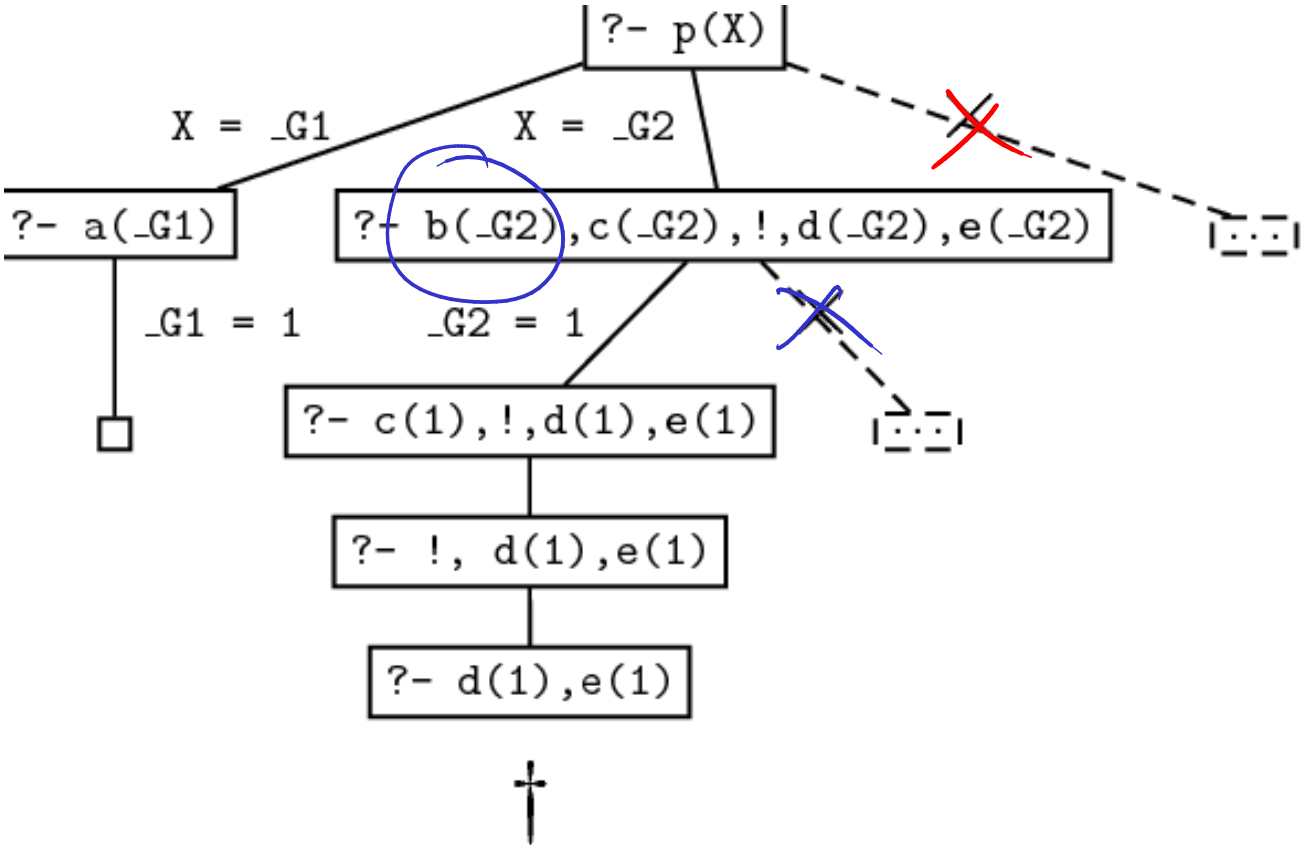
- `p(X)` unifică prima dată cu regula `p(X) :- a(X).`. Instanțiind `X` la 1, `a(X)` și `a(1)` unifică.
- `p(X)` unifică a doua oară cu regula `p(X) :- b(X), c(X), !, d(X), e(X).`. Instanțiind `X` la 1, `b(X)` și `b(1)` unifică. Deci, vom avea în continuare `c(1), !, d(1), e(1).` Faptul `c(1).` există în baza de cunoștințe. Mai rămânem cu `!, d(1), e(1).`
- Prima dată, `!` (**cut**) reușește (este satisfiabil). Nu avem în baza de cunoștințe fapte pentru `d(1), e(1)`. În concluzie, ajungem la `fail`.
- Când se revine prin backtracking la `!` (**cut**), `!` (**cut**) eșuează.
- **Toate regulile următoare cu același fel de antet (același predicat) sunt ignorate.**

Exemplul VI (cu !)

```
p(X):- a(X).
p(X):- b(X),c(X),!,d(X),e(X).
p(X):- f(X).
a(1).
```

```
b(1).
b(2).
c(1).
c(2).
```

```
d(2).
e(2).
f(3).
```

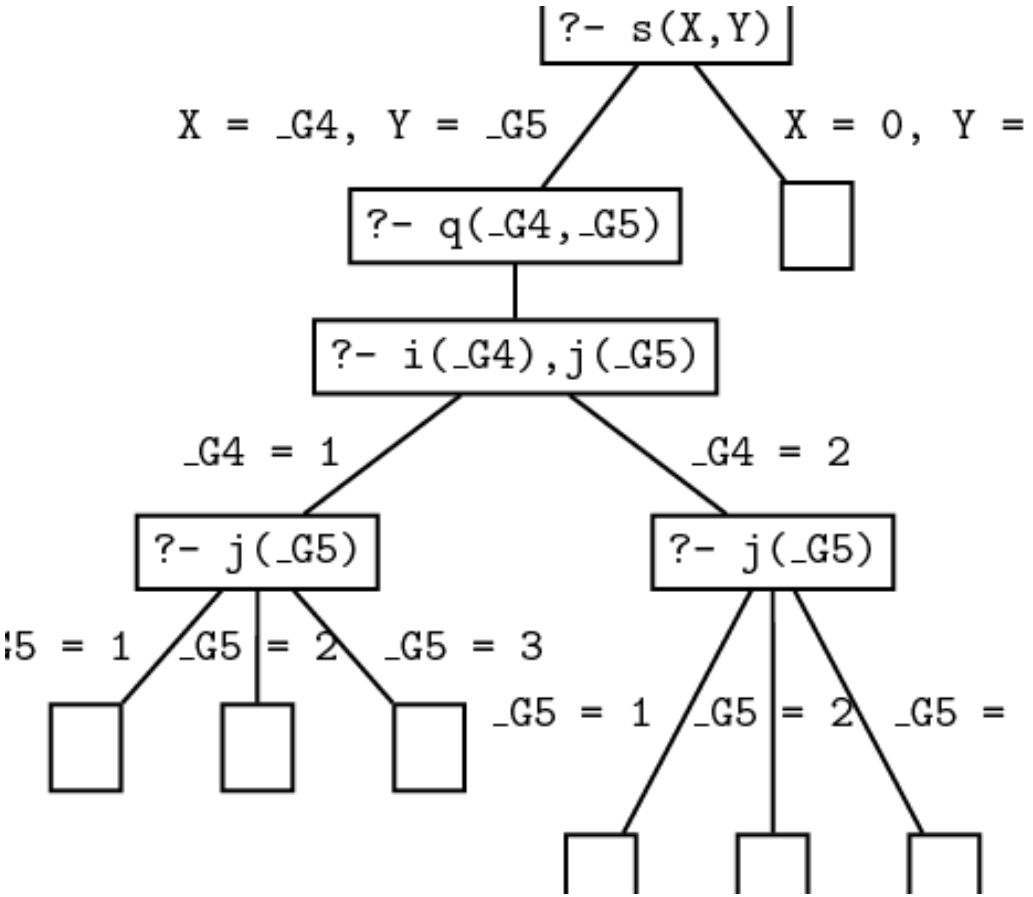


Exemplul VII

```
s(X, Y):- q(X, Y).
s(0, 0).
q(X, Y):- i(X), j(Y).
```

```
i(1).
i(2).
j(1).
```

```
j(2).
j(3).
```

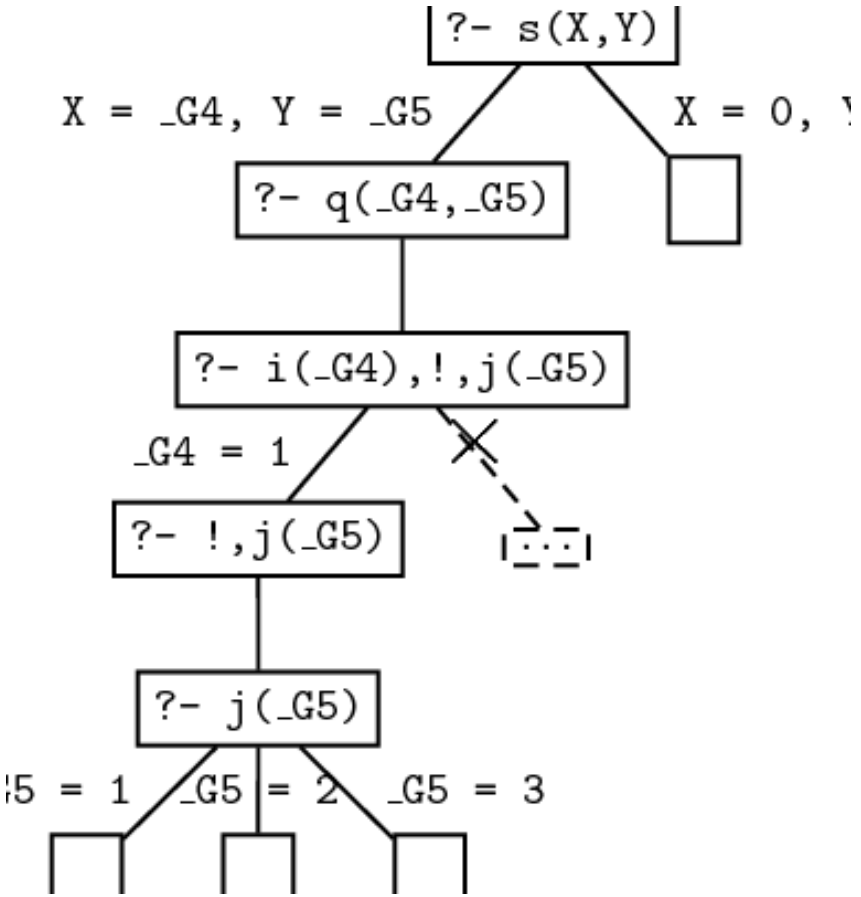


Exemplul VIII (cu !)

```
s(X, Y):- q(X, Y).
s(0, 0).
q(X, Y):- i(X), !, j(Y).
```

```
i(1).
i(2).
j(1).
```

```
j(2).
j(3).
```



Observații

- Există două contexte diferite în care putem folosi **!** (**cut**):
  1. **tăiere verde** – predicatul se introduce numai pentru creșterea eficienței programului (optimizează căutarea, evitând căutările inutile);
  2. **tăiere roșie** – modifică semnificația procedurală a programului.

```
mai_mare(X, X, X) :- !.
mai_mare(X, Y, Y) :- X < Y, !.
mai_mare(X, Y, X).
```

permutare 4 (+ L, ? Perm). L are 4 elemente

Perm = [_G1, _G2, _G3, _G4]

```
permutare4([X1,X2,X3,X4], Perm):-
length(Perm,4), member(X1, Perm),
                member(X2, Perm),
                member(X3, Perm),
                member(X4, Perm),
                length(Perm, 4).
```

?- permutare 4 ([1,2,3,4], Perm).