II COMPITO DI INTELLIGENZA ARTIFICIALE

Prof. Evelina Lamma

16 Maggio 2002 – ore 11 AULA 9

Esercizio 1

Si consideri il seguente programma Prolog:

```
c1)  complement([],L,L).
c2)  complement([H|T],L,_):-  not member(H,L), !,
                              fail.
c3)  complement([H|T],L,L2):- delete(H,L,L1),!,
                              complement(T,L1,L2).

c4)  delete(H,[H|T],T).
c5)  delete(H,[B,C|T],[B|T1]):- delete(H,[C|T],T1).
```

che realizza il complemento insiemistico (dato come secondo argomento l'insieme di tutti gli elementi), quando gli insiemi (per esempio, di interi) sono rappresentati da liste ordinate.
Si mostri l'albero SLDNF relativo all'interrogazione:

```
?-complement([1,2,4],[1,2,3,4],Y).
```

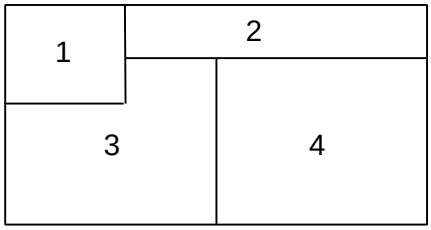
che produce la risposta:

```
yes   Y=[3]
```

Nel disegno dell'albero non si espandano i rami che vengono tagliati dal predicato cut (!), ma li si indichi soltanto.

Esercizio 2

Si abbia il seguente problema di colorazione di una mappa, in cui sono disegnate quattro regioni da colorare avendo a disposizione tre colori (individuati dai primi tre numeri naturali, 1, 2 e 3) in modo che due regioni adiacenti non siano colorate con lo stesso colore:



Il problema sia modellato in Prolog dai seguenti fatti:

```
domain([1,2,3]).        //colori disponibili

adiac(1,2).         // regione 1 e 2 adiacenti
adiac(1,3).         // regione 1 e 3 adiacenti
adiac(2,3).            ...
adiac(2,4).
adiac(3,4).
```

e sia data la seguente nozione di incompatibilità tra coppie di variabili e corrispondenti valori per il problema in esame:

```
noattack(V1,V2,C1,C2):-
```

```
    (adiac(V1,V2); adiac(V2,V1)),!, C1\==C2.
noattack(V1,V2,_,_):- not adiac (V1,V2).
```

Si realizzi un programma Prolog che generi la soluzione applicando la tecnica di standard backtracking.
In particolare, si realizzi un predicato, invocato come segue:

```
?-solution([r(1,C1), r(2,C2), r(3,C3), r(4,C4)])
```

in grado di istanziare i colori (C1, C2, ...) per le quattro regioni, considerando il dominio del problema ed applicando la tecnica di standard backtracking.

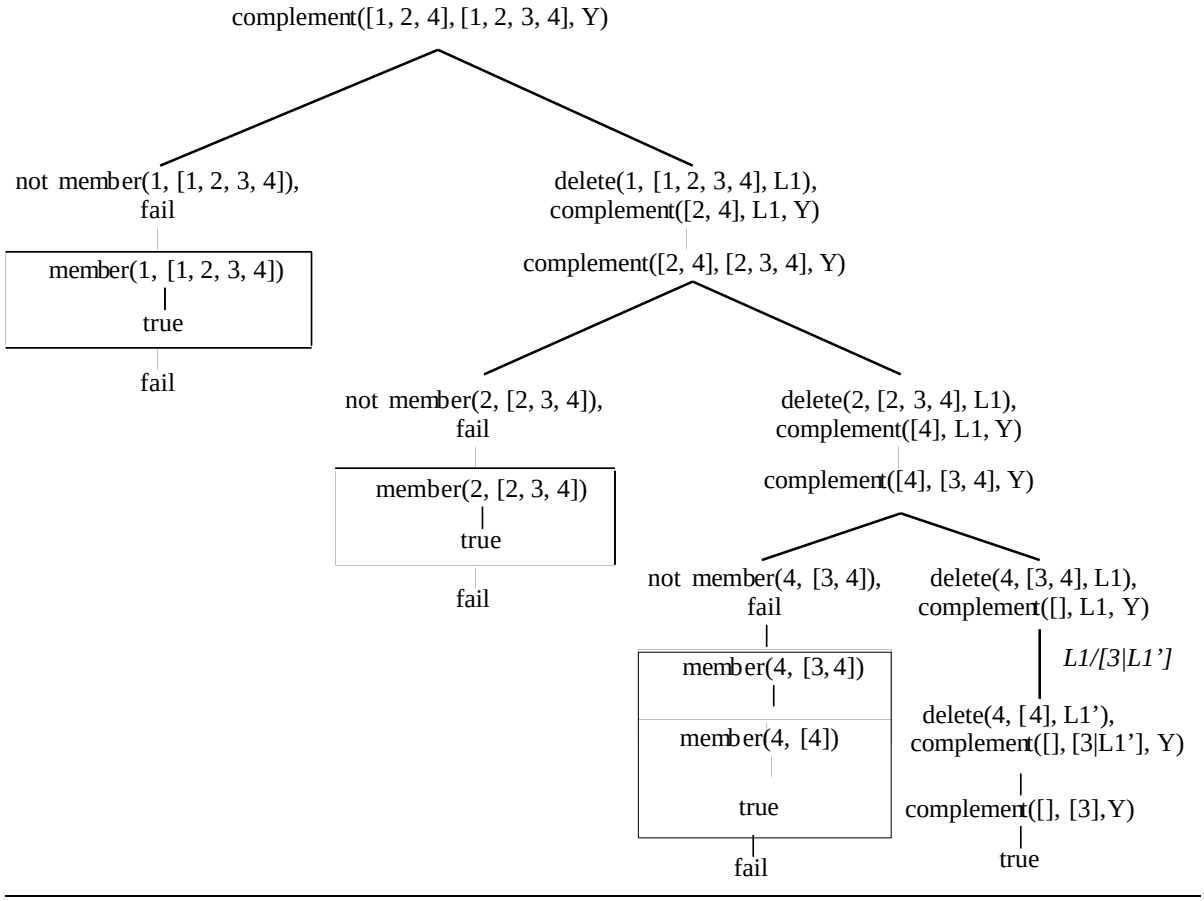
Esercizio 3 (RECUPERO I COMPITO)

Urtando una signora al mercato, un signore manda in frantumi un mucchietto di uova. Desolato, si offre di ripagarle, ma la signora non sa quante fossero le uova. Afferma, però che contando le uova a due a due ne rimaneva una, contandole tre a tre ne rimanevano due, contandole quattro a quattro ne rimanevano tre. Senz'altro, in totale, c'erano meno di venti uova.

Si formalizzi il problema come CSP, lo si renda arc-consistente (mostrando tutti i passi) e si applichi lo standard backtracking per trovare il numero di uova.

SOLUZIONE

Esercizio 1



Esercizio 2

```
solution(L):-
        domain(D),
         map(L,[],D).

map([], Placed, _).
map([r(X,Cx)|Xs], Placed, D):-
         member(Cx, D),
         noattack_placed(r(X,Cx),Placed),
         map(Xs,[ r(X,Cx),|Placed],D).

noattack_placed(r(X,Cx),[]):-!.
noattack_placed(r(X,Cx),[r(Y,Cy)|RestPlaced]):- noattack(X,Y,Cx,Cy).
```

La member/3 istanzia la variabile X ad un valore contenuto nel dominio.
La noattack_placed/2 controlla (a posteriori) che il valore scelto per X sia compatibile con quelli delle variabili già istanziate.

Esercizio 3

Formalizzazione:

Numero di uova N :: 0..20
uso tre variabili per considerare i divisori per 2,3 e 4:
D2,D3, D4 :: 0..20
Vincoli:

1. N = 2*D2+1
2. N = 3*D3+2
3. N = 4*D4+3

Arc-Consistenza:

Vincolo	**dom(N)**	**dom(D2)**	**dom(D3)**	**dom(D4)**
N = 2*D2+1	1,3,5,7,9,11,13,15,17,19	0..9	0..20	0..20
N = 3*D3+2	5,11,17	0..9	1,3,5	0..20
N = 4*D4+3	11	0..9	1,3,5	2
N = 2*D2+1	11	5	1,3,5	2
N = 3*D3+2	11	5	3	2

Standard Backtracking

Si istanziano le variabili all'unico valore nel dominio; il numero di uova è 11.