

Esercizio 3 (5 punti)

Dato il seguente programma Prolog, `rangeList(L1, L2, Min, Max)` che riporta nella lista `L2` gli elementi della lista `L1` esclusi nell'intervallo con estremi `Min` e `Max`:

```
esclusi([], [], _, _).
esclusi([A|B], [A|T], Min, Max) :-
    A <= Min, !, esclusi(B,T,Min,Max).
esclusi([A|B], [A|T], Min, Max) :-
    A >= Max, !, esclusi(B,T,Min,Max).
esclusi(_|B, T, Min, Max) :- esclusi(B,T,Min,Max).
```

disegnare l'albero SLD per il goal seguente (si indichino i tagli effettuati dal *cut* e non si espandano gli eventuali rami tagliati):

```
?- esclusi([1,2,10], L, 3, 7).
```

Esercizio 4 (5 punti)

Si consideri una lista `L` formata a sua volta da liste di numeri interi. Si scriva un programma PROLOG `aggiungi(L, B, L1)` che, data la lista di liste `L`, e il numero intero `B`, produca in uscita la lista di liste `L1` in cui ad ogni lista elemento di `L` che non contenga già `B`, è stato aggiunto il numero `B` in testa.

Ad esempio alla query:

```
?- aggiungi([[1,5],[1,3]],5, L1).
```

la risposta sarà:

```
yes L1 = [[1,5], [5,1,3]]
```

Si riportino nella soluzione le definizioni di eventuali predicati ausiliari utilizzati.

Esercizio 5 (7 punti)

“Sono arrivati in un negozio di animali quattro nuovi esemplari: un gatto, un topo, un ramarro e un serpente. Purtroppo, le gabbie sono solamente tre (denominate a, b, c). Il problema è che il gatto si mangerebbe volentieri sia il topo sia il ramarro. Il ramarro a sua volta ha paura del topo, ed il serpente, se potesse, mangerebbe il topo. Riesci a suggerire chi potrebbe andare in gabbia insieme?”

Si modelli il problema come un CSP, utilizzando quattro variabili X_i per ciascuno degli animali (nell'ordine: gatto, topo, serpente e ramarro), e si determini il valore (numero della gabbia) assegnato a ciascuna dati i vincoli. Si risolva il problema modellato con il metodo di *forward checking* mostrando i vari passi fino a trovare la prima soluzione. Si considerino le variabili secondo il loro pedice (ovvero nell'ordine dato: gatto, topo, serpente e ramarro)

Si forzi poi il backtracking cronologico e si trovi, se possibile, un'altra soluzione.

Esercizio 6 (punti 4)

Descrivere la ricerca A^* e definire sotto quali condizioni tale algoritmo di ricerca trova la soluzione ottima.

9 Gennaio 2020 - Soluzioni

Esercizio 1

1. $\forall X, \forall Y \text{ haMen}(Y) \wedge \text{haAmico}(Y,X) \rightarrow \text{vaccinaMen}(X).$
 2. $\forall X \text{ testMenPos}(X) \wedge \text{febbraAlta}(X) \rightarrow \text{hasMen}(X).$
 3. $\forall Y, \exists X \text{ haAmico}(Y, X).$
 4. $\text{febbreAlta}(\text{giovanni}).$
 5. $\text{testMenPos}(\text{giovanni}).$
- Query: $\exists X \text{ vaccinaMen}(X).$

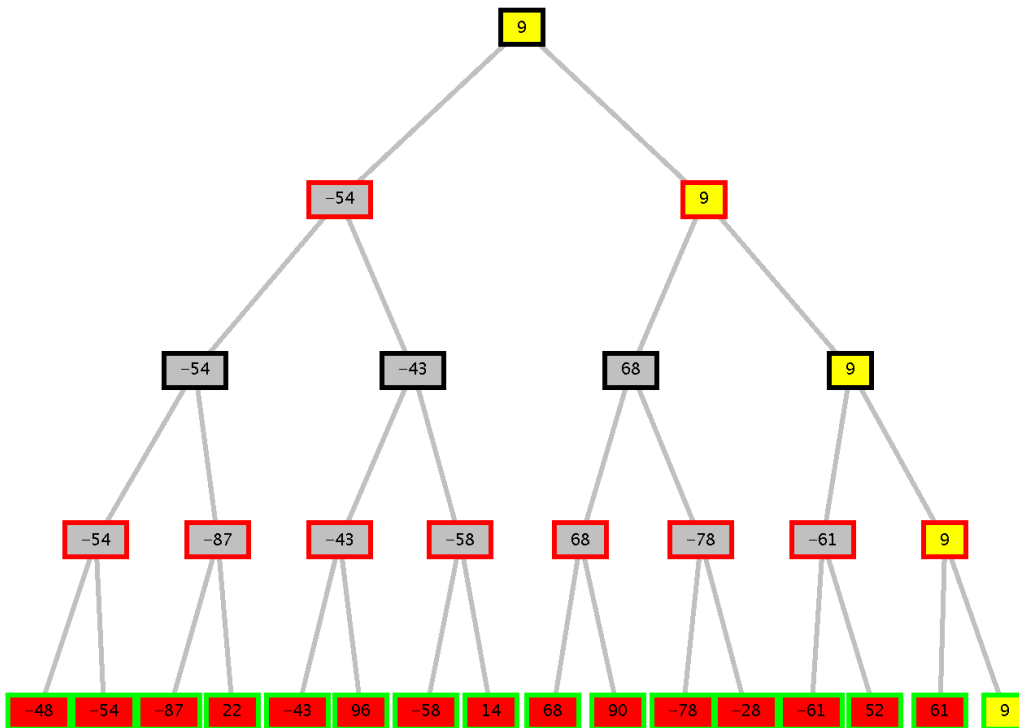
Trasformazione in clausole:

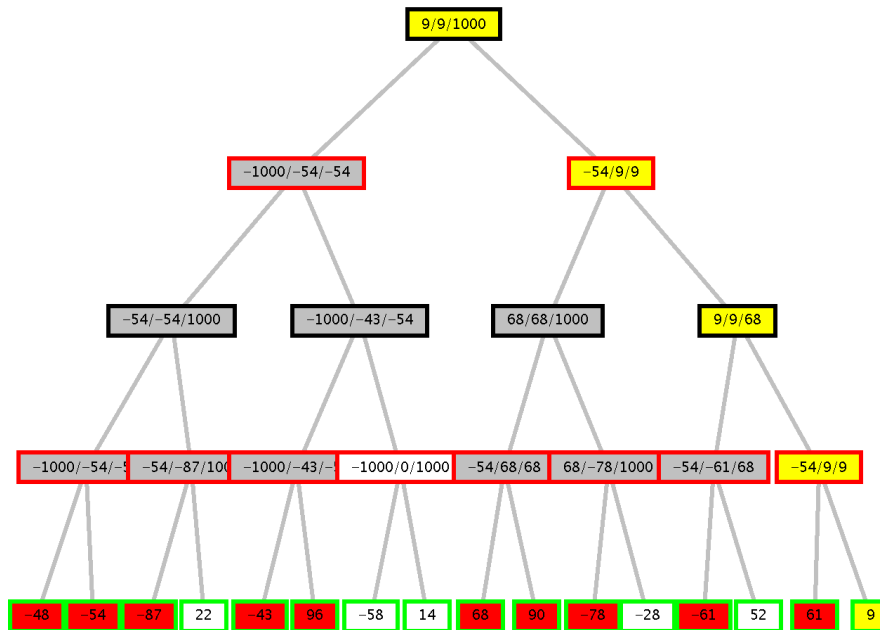
- C1: $\neg \text{haMen}(Y) \vee \neg \text{haAmico}(Y,X) \vee \text{vaccinaMen}(X).$
 C2: $\neg \text{testMenPos}(X) \vee \neg \text{febbreAlta}(X) \vee \text{hasMen}(X).$
 C3: $\text{haAmico}(Y, a(Y))$ *a(...) è la funzione di Skolem*
 C4: $\text{febbreAlta}(\text{giovanni})$
 C5: $\text{testMenPos}(\text{giovanni})$
 QNeg: $\neg \text{vaccinaMen}(X).$

Risoluzione:

- C8: QNeg+C1: $\neg \text{hasMen}(Y) \vee \neg \text{haAmico}(Y,X)$
 C9: C8 + C3: $\neg \text{hasMen}(Y)$
 C10: C9 + C2: $\neg \text{testMenPos}(X) \vee \neg \text{febbreAlta}(X)$
 C11: C10 + C4: $\neg \text{testMenPos}(\text{giovanni})$
 C12: C11 + C5: *contraddizione!*

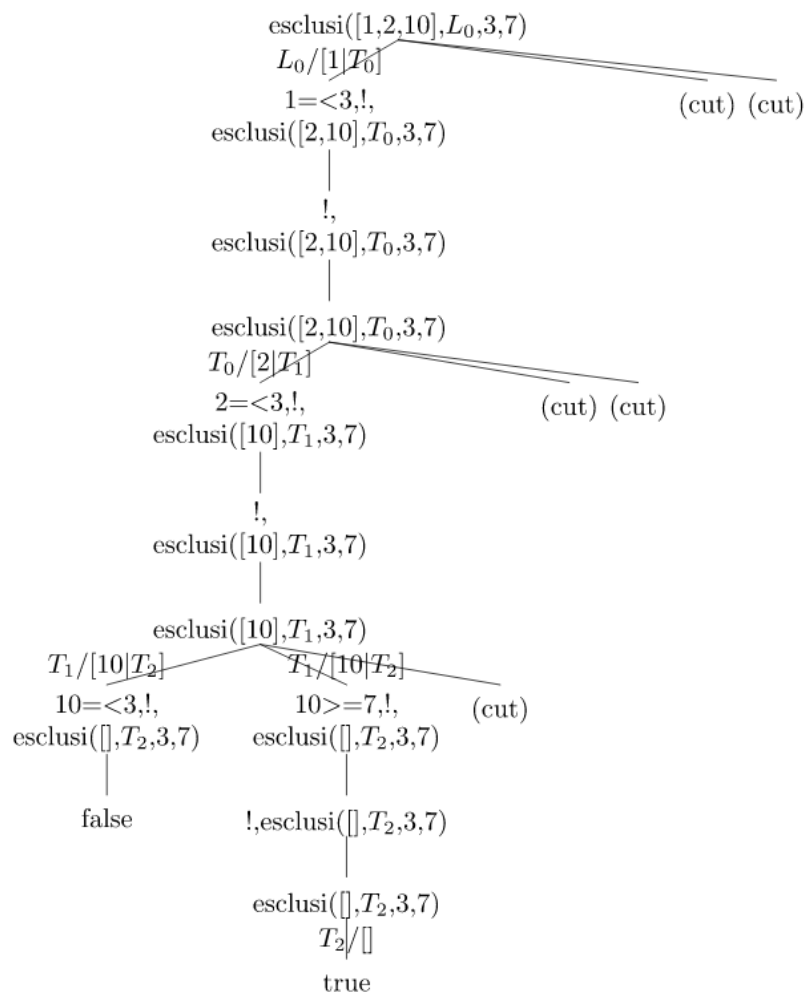
Esercizio 2





In rosso i nodi espansi, in giallo la strada trovata, i nodi in bianco non sono esplorati per effetto dei tagli alfa-beta.

Esercizio 3



Esercizio 4

```
aggiungi([], _, []).
aggiungi([H|T], B, [H|T1]) :-
    member(B, H),
    !,
    aggiungi(T, B, T1).
aggiungi([H|T], B, [[B|H]|T1]) :-
    aggiungi(T,B,T1).
```

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

Esercizio 5

Il problema è rappresentabile come un CSP a quattro variabili, ciascuna corrispondente a uno degli animali: $X_{\text{gatto}}, X_{\text{topo}}, X_{\text{serpente}}, X_{\text{ramarro}}$. Il valore assunto da ciascuna variabile indica la gabbia (che denomineremo a, b e c) in cui è possibile mettere lo specifico animale. Le quattro variabili possiedono lo stesso dominio $D_{\text{gatto}} = D_{\text{topo}} = D_{\text{serpente}} = D_{\text{ramarro}} = \{a, b, c\}$ poiché ciascun animale potrebbe essere messo in una qualsiasi delle tre gabbie (se non considerassimo i vincoli di incompatibilità). I vincoli che devono essere soddisfatti per ottenere la soluzione del quesito sono: $X_{\text{gatto}} \neq X_{\text{topo}}, X_{\text{gatto}} \neq X_{\text{ramarro}}, X_{\text{ramarro}} \neq X_{\text{topo}}, X_{\text{serpente}} \neq X_{\text{topo}}$.

Risolvendolo poi con il metodo di forward checking, i vari passi fino a trovare la prima soluzione sono i seguenti, considerando le variabili con ordine $X_{\text{gatto}}, X_{\text{topo}}, X_{\text{serpente}}, X_{\text{ramarro}}$

XG,	XT,	XS,	XR
a,b,c	a,b,c	a,b,c	a,b,c
a	b,c	a,b,c	b,c
a	b	a,c	c
a	b	a	c

Forzando poi il backtracking cronologico è possibile trovare un'altra soluzione:

XG,	XT,	XS,	XR
a,b,c	a,b,c	a,b,c	a,b,c
a	b,c	a,b,c	b,c
a	b	a,c	c (*) <i>punto di scelta più recente backtracking</i>
a	b	c	c

Esercizio 6

Vedi slide del corso.