

Esercizio 1 (6 punti)

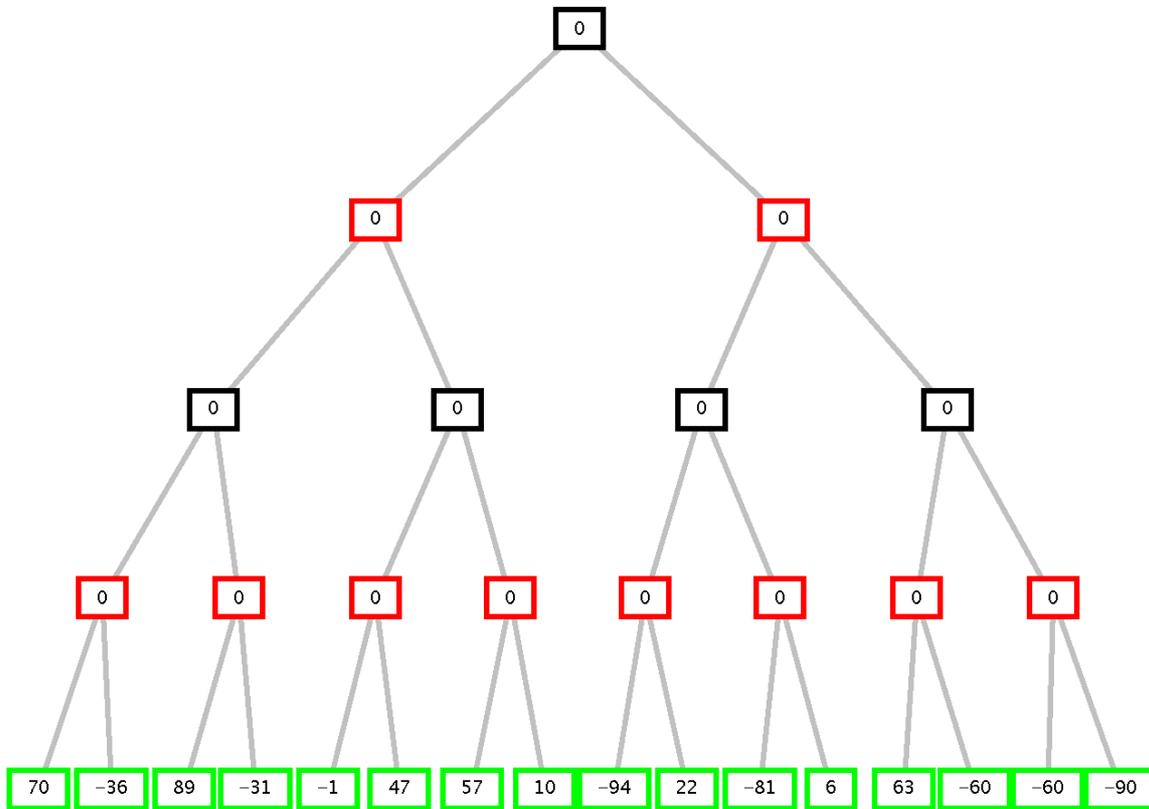
Si formalizzino le seguenti frasi in logica dei predicati:

1. I cani sono animali.
2. Qualunque cane che abbaia spaventa.
3. Qualunque animale che spaventa non è amato.
4. Esiste un cane che abbaia.

usando i seguenti predicati: **cane (X)** (X è un cane), **animale (X)** (X è un animale), **abbaia (X)** (X abbaia), **spaventa (X)** (X spaventa, è spaventoso), **amato (X)** (X è amato). Le si trasformi in clausole e si usi poi il principio di risoluzione per dimostrare che c'è un animale che non è amato.

Esercizio 2 (5 punti)

Si consideri il seguente albero di gioco in cui il primo giocatore è MAX. Si mostri come l'algoritmo *min-max* e l'algoritmo *alfa-beta* risolvono il problema e la mossa selezionata dal primo giocatore.



Esercizio 3 (6 punti)

Dato il seguente programma Prolog, aggiungi(L, B, L1) che, data la lista L, e il numero intero B, produce in uscita la lista L1 in cui ad ogni elemento minore di 0 di L è stato sommato il numero B:

```
aggiungi([],_, []).
aggiungi([H|T],B,[H1|T1]):- H<0, !, H1 is H+B,
                             aggiungi(T,B,T1).
aggiungi([H|T],B,[H|T1]):- aggiungi(T,B,T1).
```

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

```
?-aggiungi ([-11,3],5, L1).
```

Esercizio 4 (5 punti)

Si consideri una lista L formata, a sua volta, da liste di numeri interi. Scrivere un programma PROLOG `cresceSum(L)` che, data la lista di liste L , abbia successo se la somma degli elementi di ciascuna lista di L forma un ordinamento strettamente crescente.

E' dato il predicato che fa la somma degli elementi di una lista di interi, definito come segue:

```
sum([X],0) :- !.
sum([H|T], N) :- sum(T,N1), N is N1+H.
```

Ad esempio alla query:

```
?- cresceSum([[1,5],[1,7]])
```

la risposta sarà:

Yes
in quanto la somma degli elementi della prima lista [1,5] è 6 e la somma degli elementi della seconda lista [1,7] è 8 e $6 < 8$.

Mentre alla query:

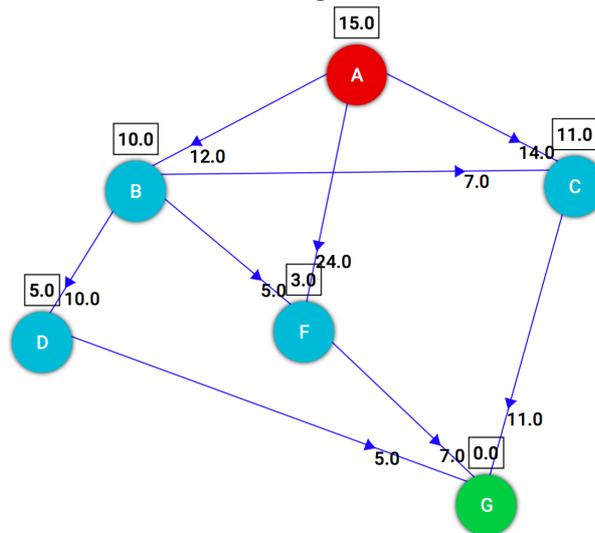
```
?- cresceSum([[1,5],[1,3]])
```

la risposta sarà:

No
in quanto la somma degli elementi della prima lista [1,5] è 6 mentre la somma degli elementi della seconda lista [1,3] è 4 e $6 > 4$.

Esercizio 5 (7 punti)

Si consideri il seguente grafo, dove A è il nodo iniziale e G il nodo goal, e il numero associato agli archi è il costo dell'operatore per andare dal nodo di partenza al nodo di arrivo dell'arco. Vicino ad ogni nodo, in un quadrato, è indicata inoltre la stima euristica della sua distanza dal nodo goal G:



- Si applichi la ricerca A^* e si disegni l'albero di ricerca sviluppato indicando per ogni nodo n l'ordine di espansione. In caso di non-determinismo, si scelgano i nodi da espandere in base all'ordine alfabetico del loro nome. Si consideri come euristica $h(n)$ quella indicata nel quadrato a fianco di ogni nodo in figura.
- L'euristica è ammissibile?
- Qual è il costo di cammino trovato da A^* ed il numero di nodi espansi per arrivare al goal G a partire dal nodo iniziale A?

Esercizio 6 (3 punti)

Si confrontino in termini di complessità spaziale, temporale e ottimalità gli algoritmi di ricerca non-informata *depth-first*, *breadth first* e ad *approfondimento iterativo*, immaginando che il costo degli archi sia sempre uguale a 1.

Esercizio 1

1. $\forall X \text{ cane}(X) \rightarrow \text{animale}(X)$
2. $\forall X \text{ cane}(X) \wedge \text{abbaia}(X) \rightarrow \text{spaventa}(X)$
3. $\forall X \text{ animale}(X) \wedge \text{spaventa}(X) \rightarrow \neg \text{amato}(X)$
4. $\exists X \text{ cane}(X) \wedge \text{abbaia}(X)$

Goal: $\exists X \text{ animale}(X) \wedge \neg \text{amato}(X)$

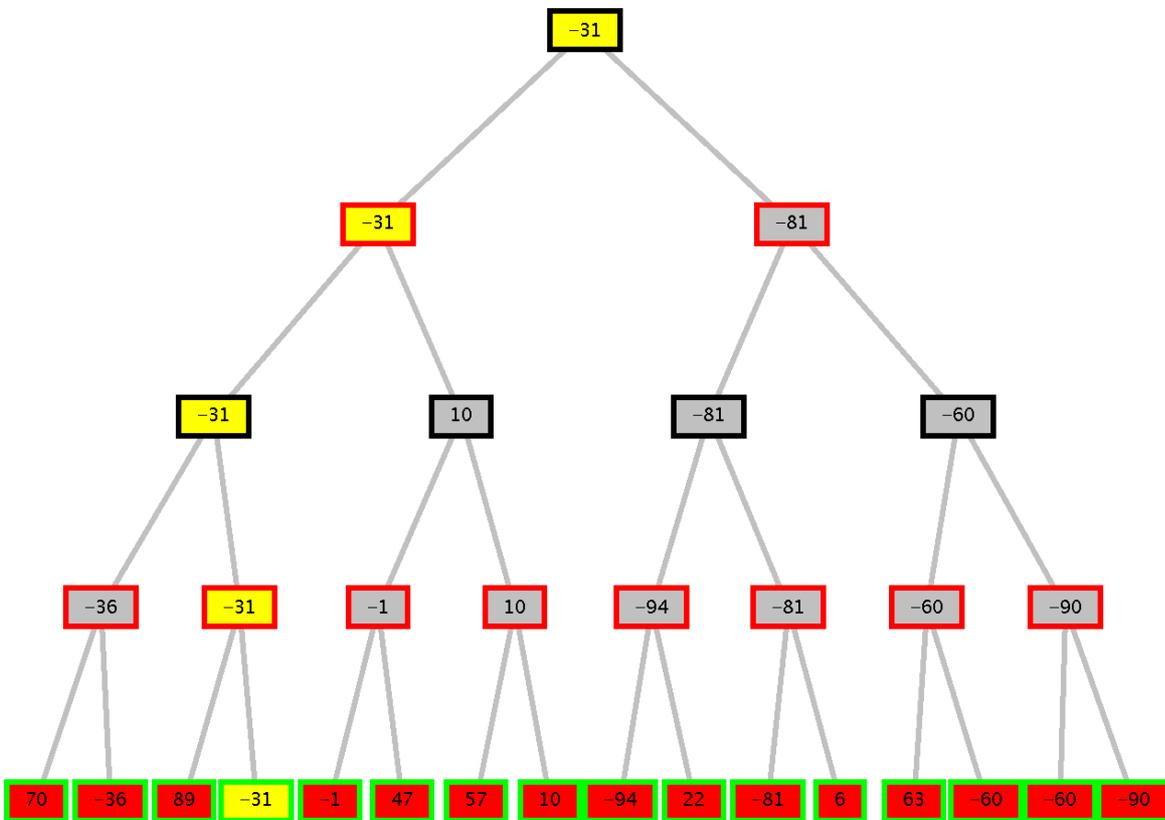
Clausole:

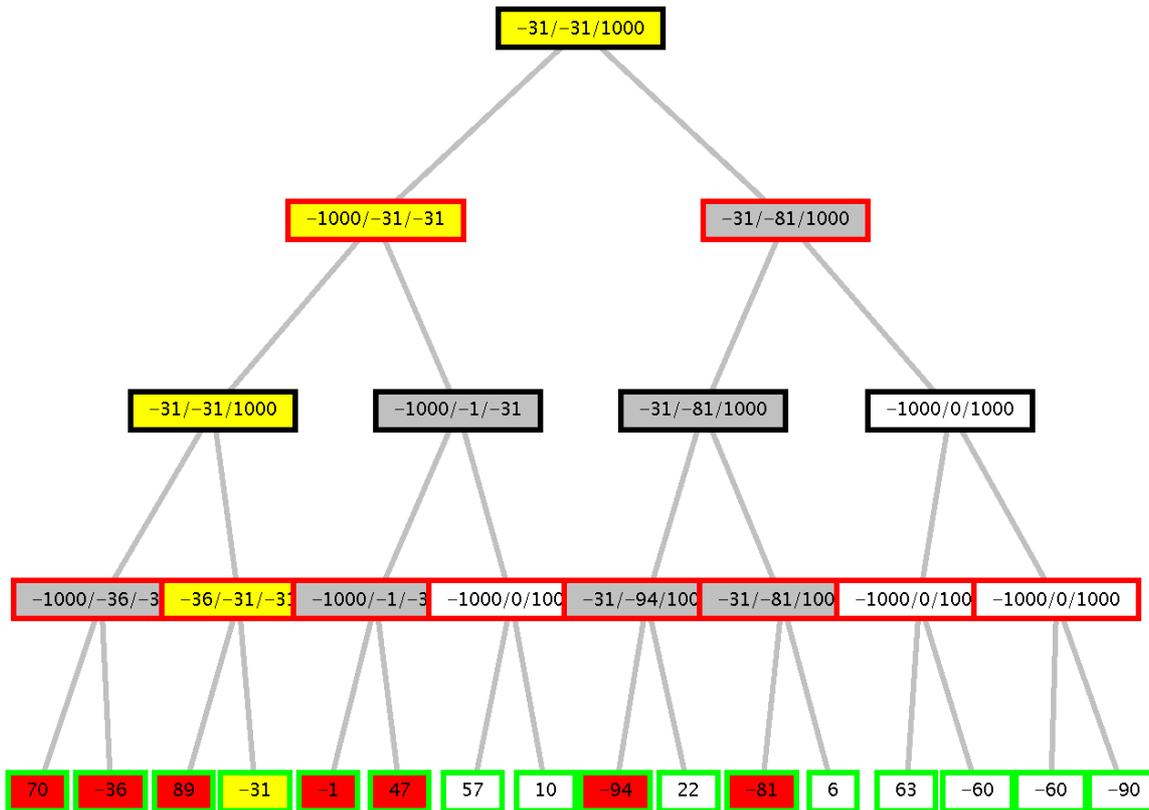
1. $\neg \text{cane}(X) \vee \text{animale}(X)$
2. $\neg \text{cane}(X) \vee \neg \text{abbaia}(X) \vee \text{spaventa}(X)$
3. $\neg \text{animale}(X) \vee \neg \text{spaventa}(X) \vee \neg \text{amato}(X)$
- 4a. $\text{cane}(c1)$ Skolem
- 4b. $\text{abbaia}(c1)$ Skolem
- GNeg: $\neg \text{animale}(X) \vee \text{amato}(X)$

Risoluzione:

- 5.: GNeg+1: $\neg \text{cane}(X) \vee \text{amato}(X)$
- 6.: 5 + 4a: $\text{amato}(c1)$
- 7.: 6 + 3: $\neg \text{animale}(c1) \vee \neg \text{spaventa}(c1)$
- 8.: 7 + 2: $\neg \text{cane}(c1) \vee \neg \text{abbaia}(c1) \vee \neg \text{animale}(c1)$
- 9.: 8 + 4b: $\neg \text{cane}(c1) \vee \neg \text{animale}(c1)$.
- 10.: 9+4a: $\neg \text{animale}(c1)$
- 11: 10+1: $\neg \text{cane}(c1)$
- 12: 11 + 4a: contraddizione!

Esercizio 2

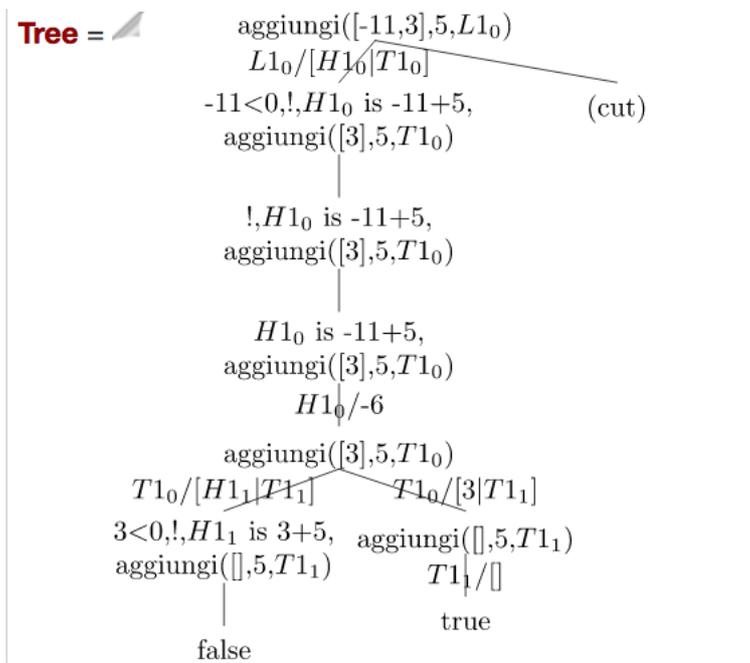




In rosso i nodi espansi, in giallo la strada trovata, i nodi in bianco non sono esplorati per effetto dei tagli alfa-beta.

Esercizio 3

Soluzione generata con <http://cplint.eu/p/mGWdAOCw.swinb>



Esercizio 4

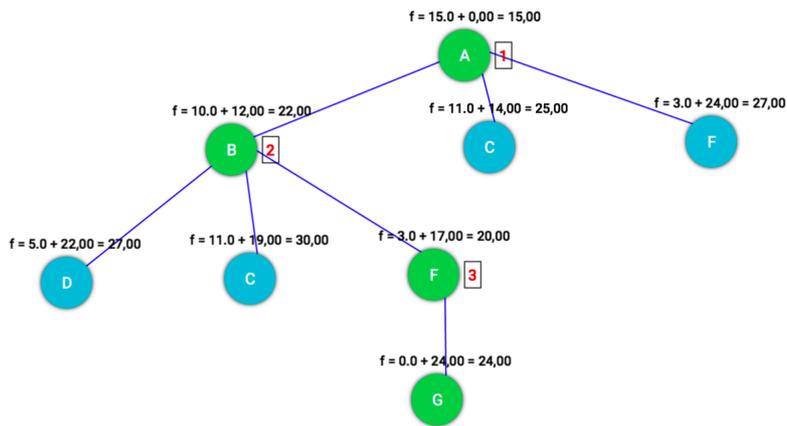
```

cresceSum([]).
cresceSum([_]).
cresceSum([H,H1|T]) :- sum(H,A), sum(H1,B), A<B, cresceSum([H1|T]).
  
```

```

sum([X],0) :- !.
sum([H|T], N) :- sum(T,N1), N is N1+H.
  
```

Esercizio 5



▼ Operations

Show operations

- 1) A [f = 15.0 + 0,00 = 15,00]
- 2) B [f = 10.0 + 12,00 = 22,00]
- 3) F [f = 3.0 + 17,00 = 20,00]
- /) D [f = 5.0 + 22,00 = 27,00]
- /) C [f = 11.0 + 19,00 = 30,00]
- /) G [f = 0.0 + 24,00 = 24,00]
- /) C [f = 11.0 + 14,00 = 25,00]
- /) F [f = 3.0 + 24,00 = 27,00]

Path cost: 24.0
Nodes expanded: 3
Queue size: 4
Max queue size: 5

L'euristica è ammissibile.

Il costo di cammino trovato da A* è 24 e il numero di nodi espansi per arrivare al goal G a partire dal nodo iniziale A è 3 (ABF), non considerando il goal test sul goal G.

Esercizio 6

Vedi materiale del corso.