FONDAMENTI DI INTELLIGENZA ARTIFICIALE

7 Settembre 2023 - Tempo a disposizione: 2 h – Risultato: 32/32 punti

Esercizio 1 (7 punti)

a) Si formalizzino le seguenti frasi in logica dei predicati del primo ordine:

Tutte le automobili che sono nell'autorimessa box1 sono più piccole delle automobili che sono nell'autorimessa box2
car1 è un'automobile.
car1 è nell'autorimessa box1 o nell'autorimessa box2. (OR esclusivo)
car2 è un'automobile.
car2 è nell'autorimessa box1
car2 non è più piccola di car1.

utilizzando i predicati:

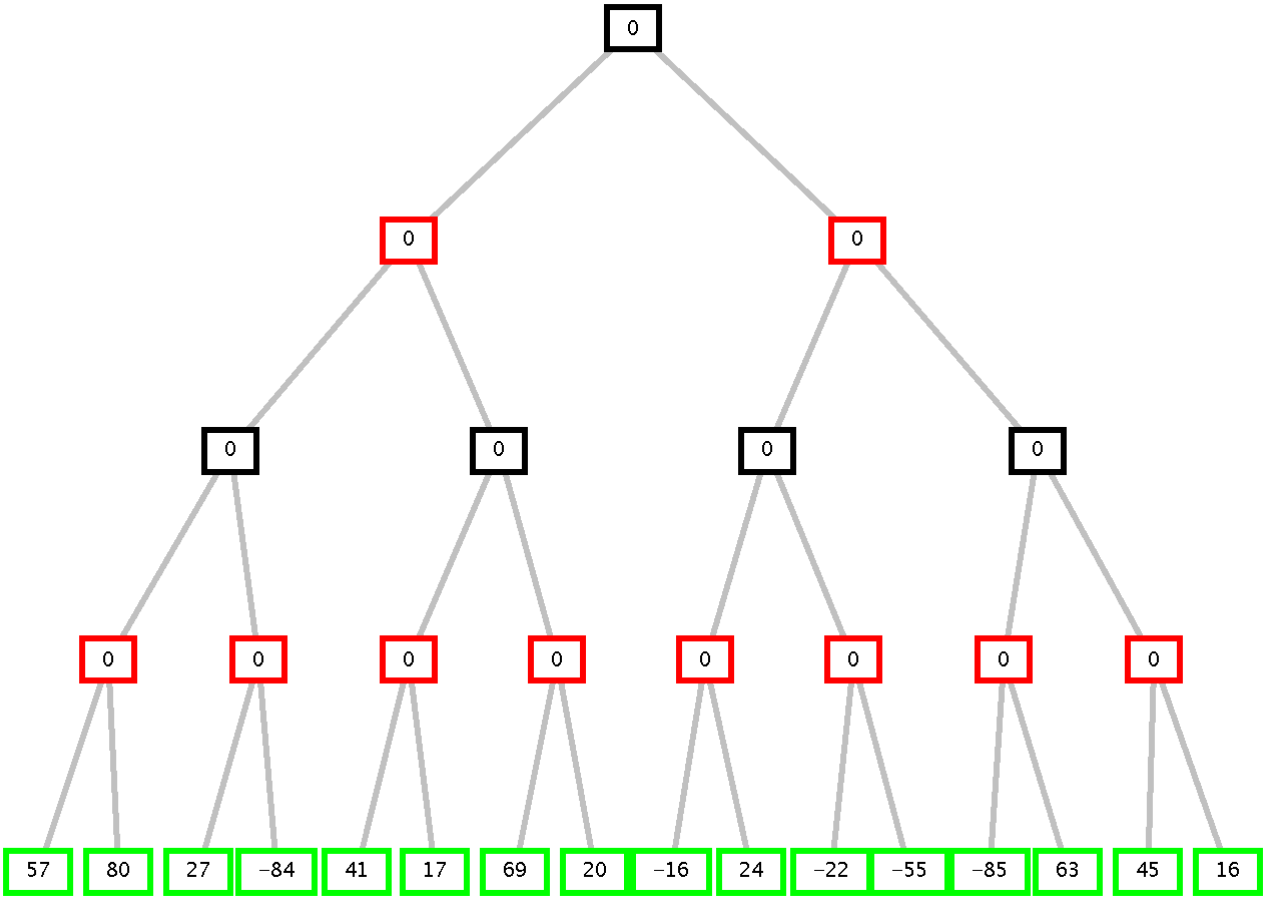
auto(*X*): X è un'automobile
inAutorimessa(*X,Box*): X è nell'autorimessa Box
piccolo(X,Y): X è più piccolo di Y

Si trasformino poi le formule in clausole, e si dimostri, applicando il principio di risoluzione, che: *car1 è nell'autorimessa box1.*

b) La base di conoscenza può essere formalizzata in clausole definite? Qual è la loro definizione? Si motivi la risposta data.

Esercizio 2 (5 punti)

Si consideri il seguente albero di gioco in cui il primo giocatore è MAX.



a) Si indichi come l'algoritmo min-max risolve il problema indicando il valore con cui viene etichettato il nodo iniziale e la mossa selezionata dal primo giocatore (arco a sinistra o a destra).
b) Si mostrino poi i tagli che l'algoritmo alfa-beta consente, indicando gli archi che verranno tagliati.

Esercizio 3 (5 punti)
Si scriva in Prolog il predicato **newList(L1, L)** che data una lista **L1** di liste **EL** composte da elementi interi oppure vuote, restituisca una nuova lista **L**, composta inserendo in **L** i valori massimi delle liste **EL**. Se un elemento della lista **L1** è la lista vuota non andrà inserito in **L**. Se la lista **L1** è vuota anche la lista **L** sarà vuota.
Si consiglia di definire un predicato ausiliario **max(L,M)** che data una lista **L** di interi restituisce in **M** il maggiore.

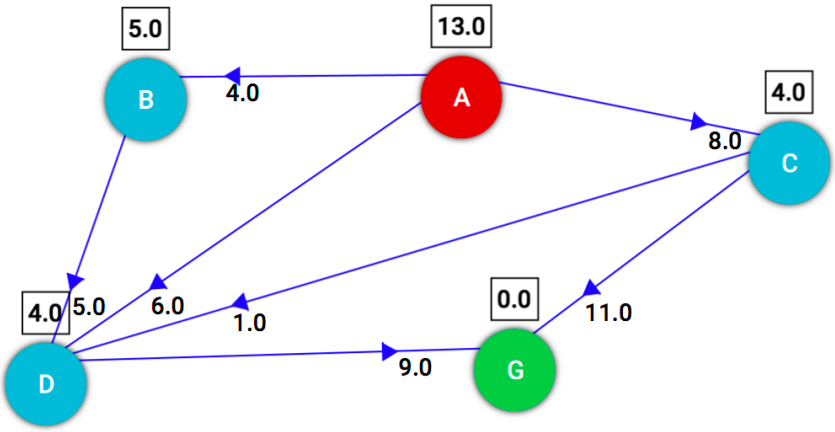
```
?-newList([[3,7,2],[11,9,4], [], [12,3], [5,5]],L).
L = [7, 11, 12, 5]
?- newList([],L).
L = []
```

Esercizio 4 (6 punti)
Dato il seguente programma Prolog:

```
newList([],[]).
newList([[]|T],T1):-newList(T,T1).
newList([[X,Y]|T],[X|T1]):- X>=Y,!, newList(T,T1).
newList([[_,Y]|T],[Y|T1]):- newList(T,T1).
```

Si mostri l'*albero SLD* relativo al goal: **?- newList([[3,7],[11,9], []],L).** Si indichi la risposta calcolata per la variabile **L** del goal.

Esercizio 5 (5 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*** su alberi (non tenendo quindi traccia dei nodi già visitati che non vengono automaticamente eliminati) **disegnando l'albero generato dinamicamente.** In caso di non determinismo si selezionino i nodi da espandere secondo l'ordine alfabetico. Si indichino:

- i nodi espansi nell'ordine di espansione;
- i nodi sulla strada della soluzione e il costo della soluzione;
- la condizione sulla stima euristica h(n) che garantisce l'ottimalità della ricerca su alberi e se è soddisfatta in questo caso.

Esercizio 6 (4 punti)
Dopo avere brevemente introdotto l'algoritmo di Forward Checking, se ne mostri l'esecuzione su questo esempio:

A::[2, 3, 4, 5, 6, 7, 8]	C::[2, 3, 4, 5, 6, 7, 8]
B::[2, 3, 4, 5, 6, 7, 8]	D::[2, 3, 4, 5, 6, 7, 8]
A<=20-B	C=2+D
C<=10-B	D>=3*A

considerando la prossima variabile da istanziare secondo l'euristica Minimum Remaining Values (MRV); a parità di MRV, si scelga la variabile che viene prima in ordine alfabetico. Nella scelta dei valori, si prediliga sempre di assegnare il valore minore. Si mostri la riduzione dei domini delle variabili future dopo ogni labeling, fino alla prima soluzione.

7 Settembre 2023 - Soluzioni

Esercizio 1

a)

Traduzione in logica

1) ∀X ∀Y auto(X) AND auto(Y) AND inAutorimessa(X,box1) AND inAutorimessa(Y,box2) ⟹ piccolo(X,Y).
2) auto(car1)
3) inAutorimessa(car1,box1) OR-EX inAutorimessa(car1,box2)
4) auto(car2)
5) inAutorimessa(car2,box1)
6) ¬ piccolo(car2,car1).

Goal negato: ¬ inAutorimessa(car1,box1).

Trasformazione in clausole

1) ¬auto(X) OR ¬ auto(Y) OR ¬inAutorimessa(X,box1) OR ¬ inAutorimessa(Y,box2) OR piccolo(X,Y).
2) auto(car1)
3a) inAutorimessa(car1,box1) OR inAutorimessa(car1,box2)
3b) ¬ inAutorimessa(car1,box1) OR ¬ inAutorimessa(car1,box2)
4) auto(car2)
5) inAutorimessa(car2,box1)
6) ¬ piccolo(car2,car1).

GNeg: ¬ inAutorimessa(car1,box1).

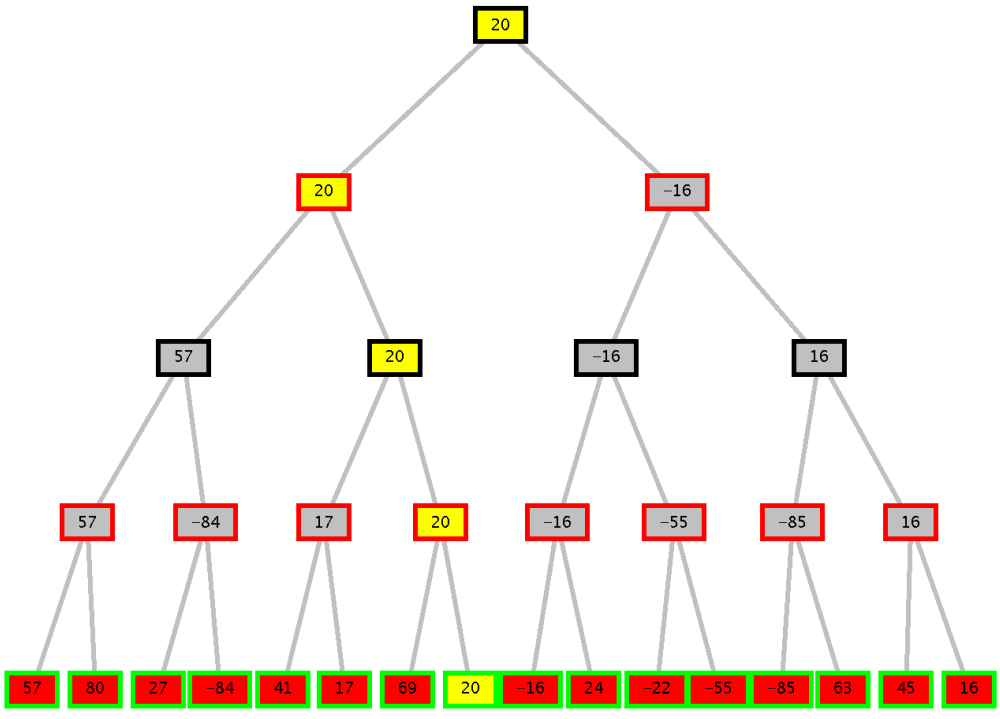
Risoluzione

7: GNeg + 3a: inAutorimessa(car1,box2).
8: 6 + 1: ¬auto(car2) OR ¬ auto(car1) OR ¬inAutorimessa(car2,box1) OR ¬inAutorimessa(car1,box2)
9: 8 + 2: ¬auto(car2) OR ¬inAutorimessa(car2,box1) OR ¬inAutorimessa(car1,box2)
10: 9 + 4: ¬inAutorimessa(car2,box1) OR ¬ inAutorimessa(car1,box2)
11: 10 + 5: ¬ inAutorimessa(car1,box2)
12: 11 + 7: contraddizione!!

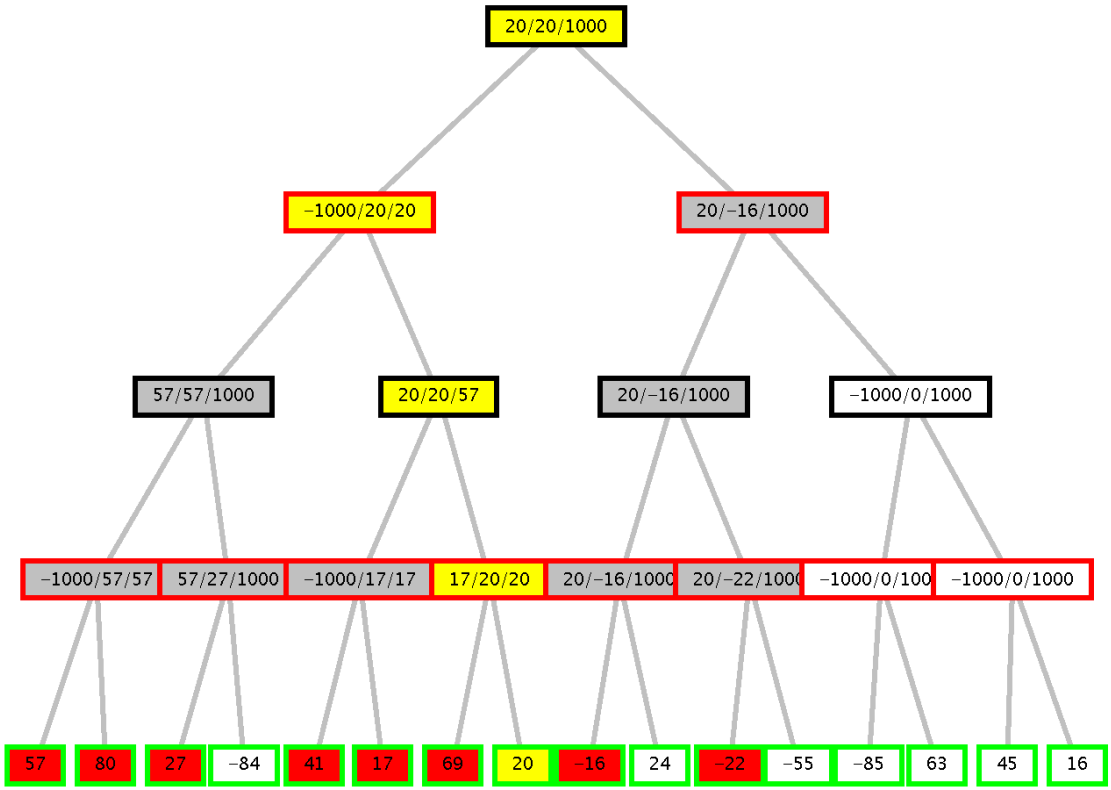
b) La base di conoscenza non può essere formalizzata in clausole definite (clausole che contengono un solo letterale positivo) a causa della clausole generali (3a), (3b), e (6), che non rispettano tale vincolo.

Esercizio 2

Min max:



Alfa Beta:



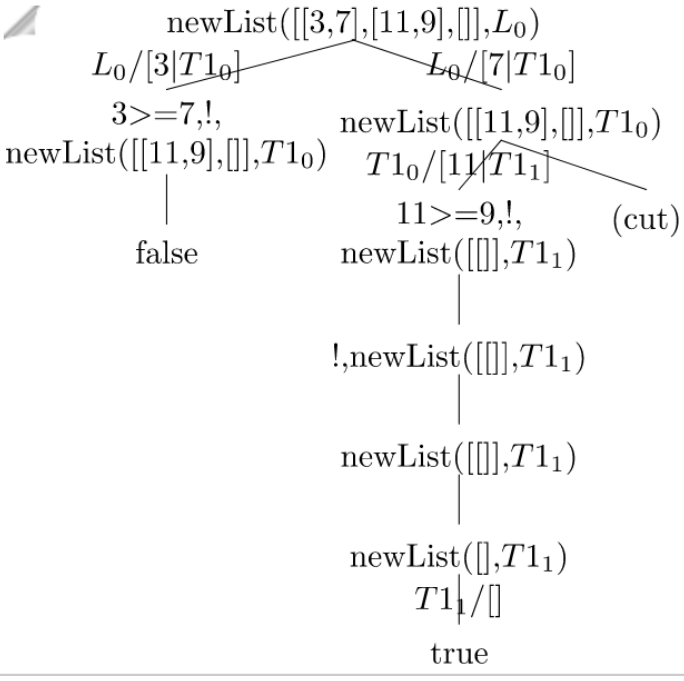
Esercizio 3

```
newList([],[]).
newList([[]|T],T1):- !, newList(T,T1).
newList([EL|T],[M|T1]):- max(EL,M),newList(T,T1).

max([X],X).
max([X|T],X):-max(T,E1), X>E1,!.
max([_|T],E1):-max(T,E1).
```

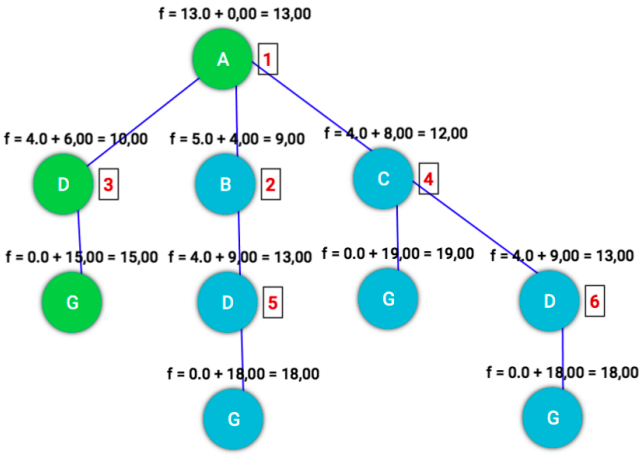
Esercizio 4

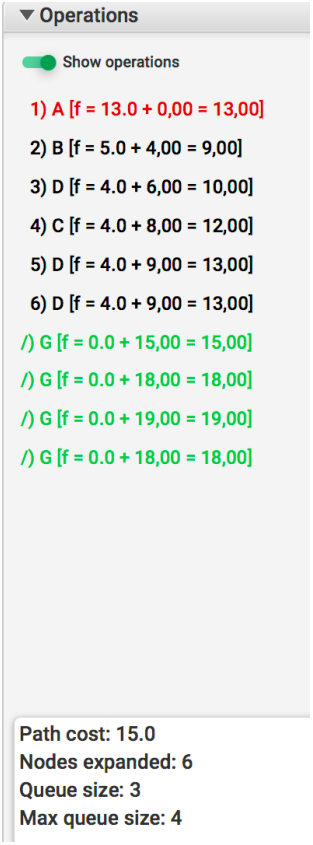
L = [7,11]



Esercizio 5

Con A* i nodi espansi sono ABDCDDG e la soluzione ADG ha costo 15 (ottimale):





La condizione sulla funzione euristica stimata h* (n) che garantisce l'ottimalità della ricerca è la condizione di ammissibilità che deve valere per ogni nodo dell'albero e che è verificata se la h*(n) è ottimista cioè h*(n) <= h(n). Tale condizione è soddisfatta in questo caso.

Esercizio 6

Vedi slide del corso per spiegare FC.

	A	B	C	D
Labeling	A=2	[2...8]	[2...8]	[2...8]
FC		[2...8]	[2...8]	[6...8]
Labeling		[2...8]	[2...8]	D=6
FC		[2...8]	[8]	
Labeling		[2...8]	C=8	
FC		[2]		
Labeling e FC		B=2		

Non c'è backtracking.