FONDAMENTI DI INTELLIGENZA ARTIFICIALE

21 Giugno 2024 - Tempo a disposizione: 2 h – Risultato: 32/32 punti

Esercizio 1 (6 punti)

Si formalizzino le seguenti frasi in logica dei predicati del I ordine:

1. *Qualunque piatto, se Giovanni lo prepara allora lo prepara anche Rosa (Se Giovanni prepara un piatto allora lo prepara anche Rosa)*
2. *...e viceversa (Se Rosa prepara un piatto allora lo prepara anche Giovanni)*
3. *Chiunque prepara un qualunque piatto complicato, allora è un cuoco.*
4. *Esiste un piatto complicato che Giovanni prepara.*

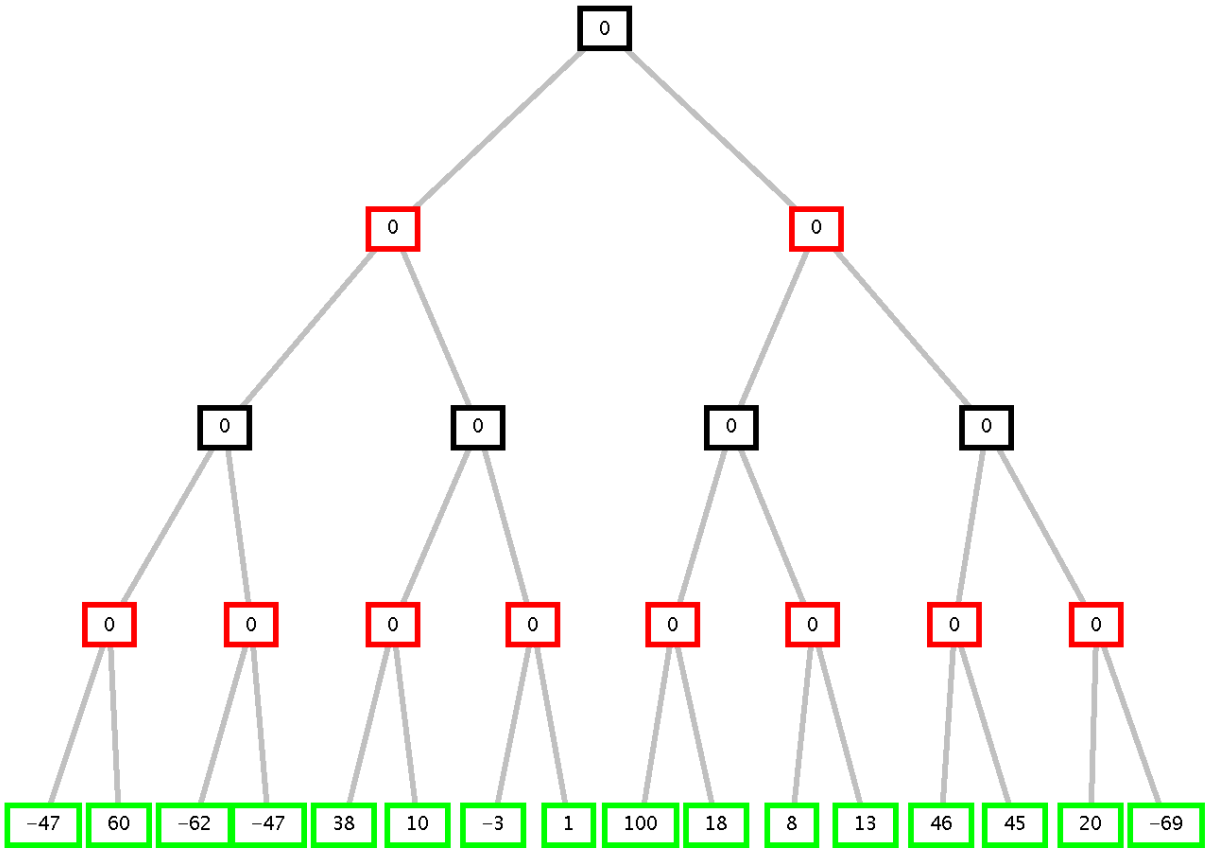
Le si trasformi in clausole e si usi la risoluzione per dimostrare che (query): *Rosa è una cuoca.*

Si usino, a tal scopo, i seguenti predicati (da significato inteso): **piatto(X)**, *X è un piatto*; **prepara(X,Y)**, *X prepara il piatto Y*; **complicato(X)**, *X è complicato*; **cuoco(X)**, *X è una cuoca/o*.

e le costanti: rosa, giovanni.

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 **maggioriLista(N,L1,L2)**, che dato il numero intero **N** e una lista **L1** di numeri interi, restituisca la lista **L2** contenente solo gli elementi della lista **L1** strettamente maggiori di **N**. Se **L1** è la lista vuota verrà restituita in uscita la lista vuota.

Esempio:

?- maggioriLista(5,[4,6,7,3], X).

X = [6, 7]

Esercizio 4 (5 punti)

Dato il seguente predicato Prolog **sostLista(L1,N,Val,L2)**, che dato il numero intero **N** e una lista **L1** di numeri interi, restituisce in **L2** la lista ottenuta sostituendo l'elemento in posizione **N** in **L1** con l'elemento **Val** (Il primo elemento di **L1** è considerato in posizione **0**. Se la posizione specificata è **>=** della dimensione di **L1** non verrà fatta alcuna sostituzione e la lista in uscita sarà quindi uguale a **L1.** Se **L1** è la lista vuota verrà restituita in uscita la lista vuota):

sostLista([],_,_,[]).
sostLista([H|T],0,X,[X|T]):-!.
sostLista([H|T],N,X,[H|Y]):-N>0,N1 is N-1,sostLista(T,N1,X,Y).

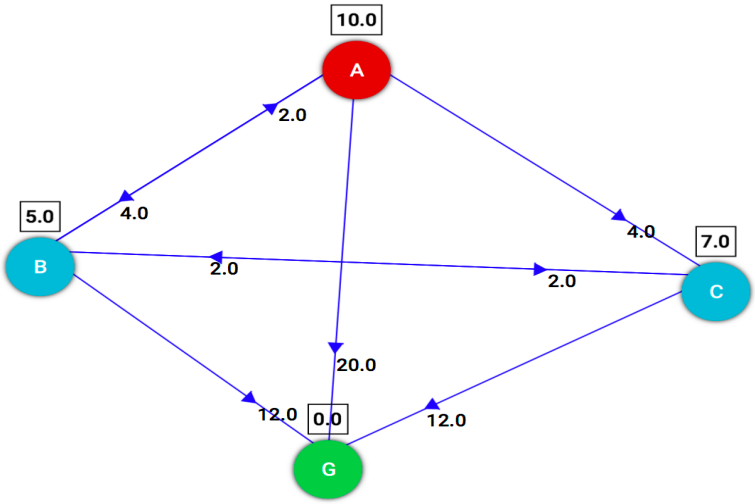
si mostri l'albero SLD generato dal goal:

?- sostLista([1,3,5,7], 2, 9,L).

indicando eventuali rami di fallimento e quelli tagliati da cut, e la risposta calcolata per la variabile **L.**

Esercizio 5 (6 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;

Si indichi per la ricerca **A*** la condizione sulla stima euristica h(n) che garantisce l'ottimalità di A* su alberi e se è soddisfatta in questo caso.

Si mostri poi cosa si ottiene applicando la ricerca **Greedy Best-first**.

Esercizio 6 (5 punti)

Dopo avere brevemente introdotto l'algoritmo di Arc-Consistency, si disegni il grafo per questo CSP e si faccia vedere l'applicazione di AC sul problema in esame.

A::[2, 3, 4, 5, 6]

B::[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

C::[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

D::[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

A>B+1

C>B-2

A=D+1

21 Giugno 2024 - Soluzioni

Esercizio 1

Rappresentazione in FOL (e in clausole):

1. Se Giovanni prepara un piatto allora lo prepara anche Rosa

∀X piatto(X) ∧ prepara(giovanni,X) ⇒ prepara(rosa,X)

∼piatto(X) ∨ ∼prepara(giovanni,X) ∨ prepara(rosa,X) C1

2. Se Rosa prepara un piatto allora lo prepara anche Giovanni

∀X piatto(X) ∧ prepara(rosa,X) ⇒ prepara(giovanni,X)

∼piatto(X) ∨ ∼prepara(rosa,X) ∨ prepara(giovanni,X) C2

3. Chiunque prepara un qualunque piatto complicato, allora è un cuoco.

∀X∀P piatto(P) ∧ prepara(X,P) ∧ complicato(P) ⇒ cuoco(X)

∼piatto(P) ∨ ∼prepara(X,P) ∨ ∼complicato(P) ∨ cuoco(X) C3

4.Esiste un piatto complicato che Giovanni prepara.

∃P piatto(P) ∧ prepara(giovanni,P) ∧ complicato(P).

k1 costante di Skolem.

piatto(k1). C4.1

prepara(giovanni,k1) C4.2

complicato(k1) C4.3

Goal: Rosa è una cuoca.

cuoco(rosa)

Gneg: ∼cuoco(rosa)

Risoluzione:

C5: Gneg + C3: ∼piatto(P) ∨ ∼prepara(rosa,P) ∨ ∼complicato(P)

C6: C5 + C4.1: ∼prepara(rosa,k1) ∨ ∼complicato(k1)

C7: C6 + C4.3: ∼prepara(rosa,k1)

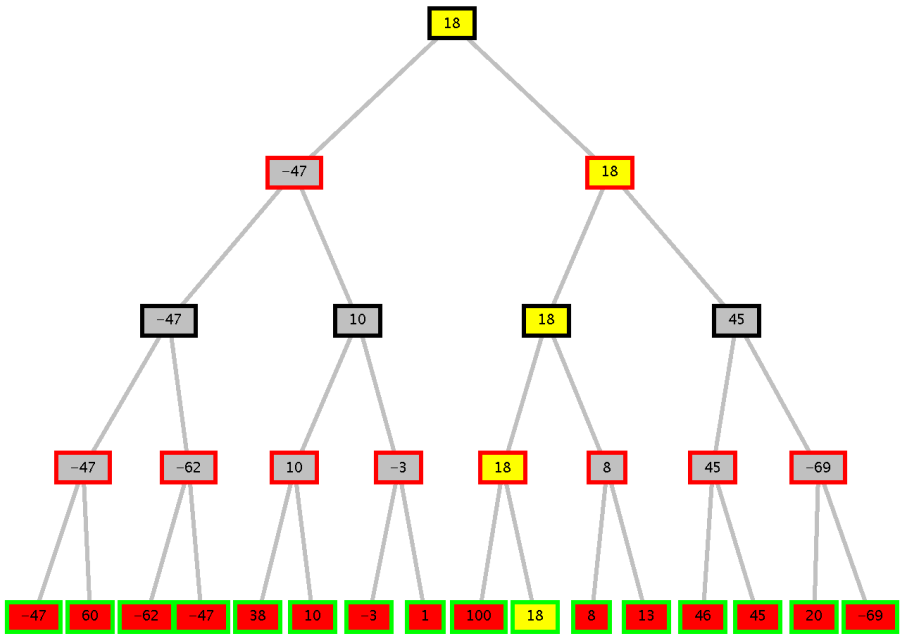
C8: C7+C1: ∼piatto(k1) ∨ ∼prepara(giovanni,k1)

C9: C8 + C4.2: ∼piatto(k1)

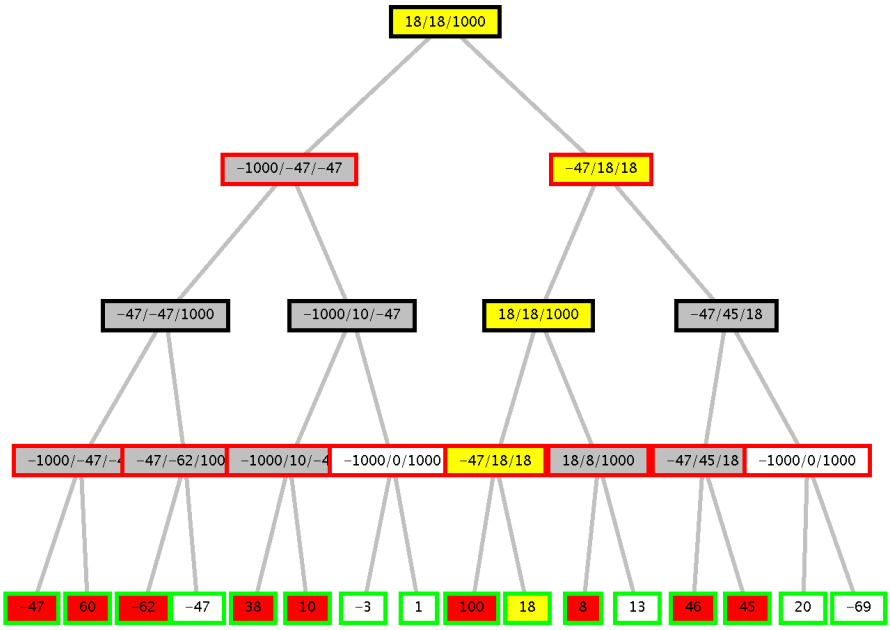
C10: C9 + C4.1: clausola vuota, contraddizione.

Esercizio 2

Min max:



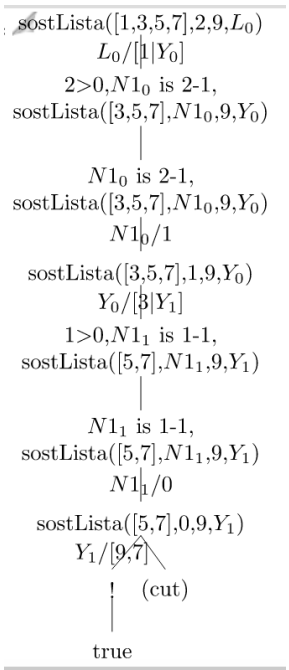
Alfa Beta:



Esercizio 3

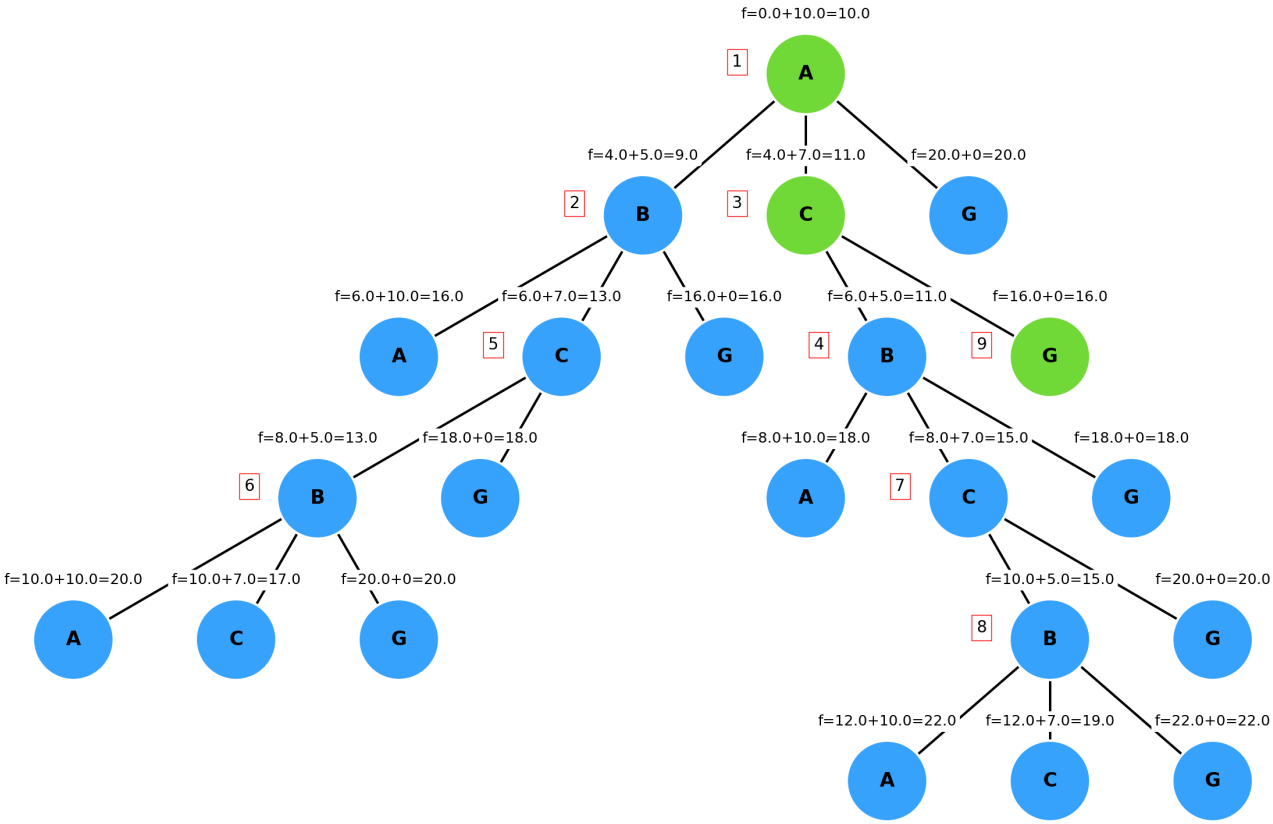
```
maggioriLista (_,[],[]).
maggioriLista (N,[H|T],[H|U]):- H>N, !, maggioriLista (N,T,U).
maggioriLista (N,[_|T],U):- maggioriLista (N,T,U).
```

Esercizio 4



Risposta calcolata: L=[1,3,9,7]

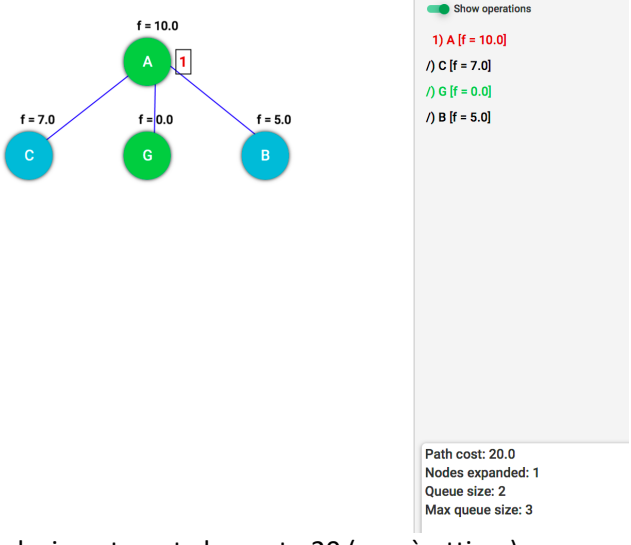
Esercizio 5
Con A*:



- I nodi espansi nell'ordine di espansione sono: ABCBCBCBABG
- I nodi sulla strada della soluzione sono ABG o ACG e il costo della soluzione è 16;

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.

Greedy best-first:



I nodi sulla soluzione sono AG, la soluzione trovata ha costo 20 (non è ottima).

Esercizio 6
Vedi slide del corso per spiegare Arc-Consistency.

	A	**B**	**C**	**D**
1 iteraz.	[2, 3, 4, 5, 6]	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
A>B+1	**[3, 4, 5, 6]**			
B<A-1		**[1, 2, 3, 4]**		
C>B-2				
B<C+2				
A=D+1				
D=A-1				**[2, 3, 4, 5]**
2 iteraz.	[3, 4, 5, 6]	[1, 2, 3, 4]	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]	[2, 3, 4, 5]
A>B+1				
B<A-1				
C>B-2				
B<C+2				
A=D+1				
D=A-1				

Nessuna ulteriore cancellazione dai domini.