FONDAMENTI DI INTELLIGENZA ARTIFICIALE

11 febbraio 2021 – Tempo a disposizione: 2 h – Risultato: 32/32 punti

NOTA: Consegnare la soluzione tramite un singolo file, che lo studente avrà cura di nominare come: CognomeNomeDataAI Ad esempio: RossiMario20210211AI

Esercizio 1 (6 punti)

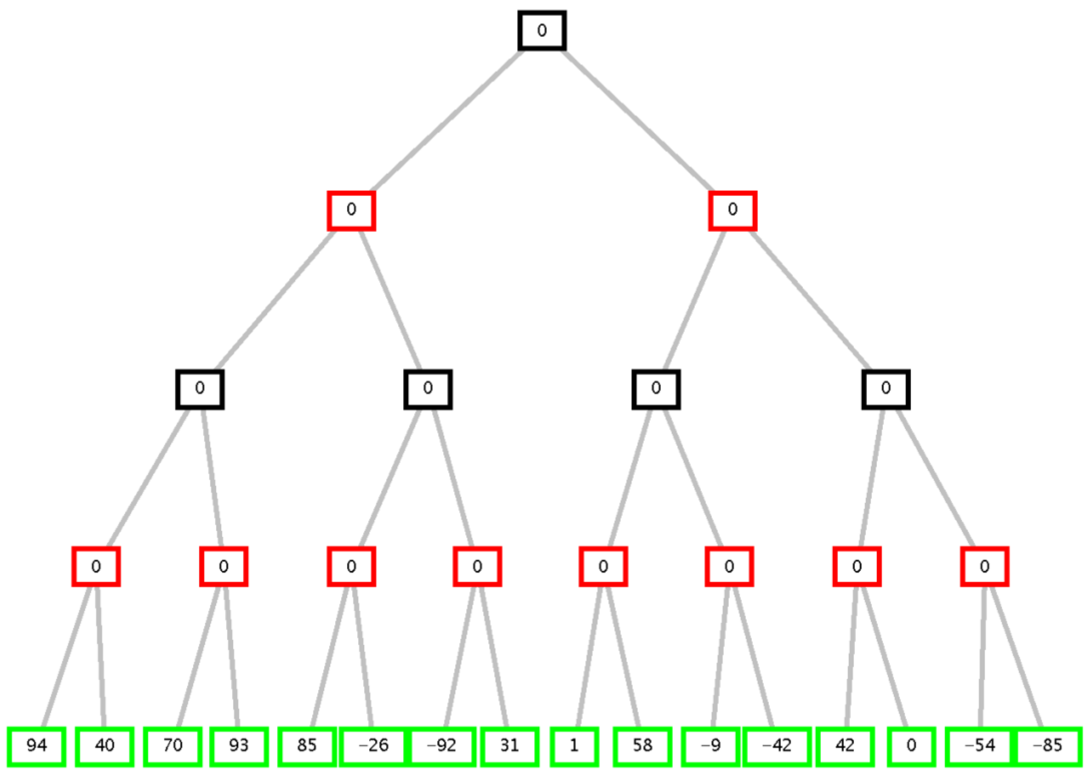
Si esprimano in logica dei predicati del I ordine le seguenti frasi:

1. Se una persona aiuta persone bisognose allora è buona.
2. Se una persona non ha un alloggio allora è bisognosa.
3. Giorgio è una persona e non ha un alloggio.
4. Maria è una persona.
5. Maria aiuta Giorgio.

Le si trasformi in clausole, e si dimostri poi, usando la risoluzione, che *Maria è buona*.
A tal scopo si usino i seguenti predicati: **buona(X)** indica che X è buona, **aiuta(X,Y)** indica che X aiuta Y; **haAlloggio**(X) indica che X ha un alloggio, **bisognosa(X)** indica che X è bisognosa, **persona (X)** indica che X è una persona.

Esercizio 2 (4 punti)

Si consideri il seguente albero di gioco in cui il primo giocatore è MAX. 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. Si mostrino poi i tagli che l'algoritmo alfa-beta consente indicando gli archi che verranno tagliati. Si indichino i nomi degli archi iniziando con la lettera "**a**" e facendola seguire con un numero crescente da sinistra a destra e dall'alto al basso. Ad esempio, i due archi che si dipartono dalla radice saranno nominati a1 (quello più a sinistra) e a2. L'arco che connette il nodo foglia più a sinistra (con valore 94) sarà denominato a15, mentre l'ultimo arco che connette il nodo foglia più a destra (valore -85) a30.



Esercizio 3 (6 punti)

Si consideri il seguente CSP che lega le variabili A, B, C, D:

A::[1, 2, 3, 4, 5, 6]
B::[1, 2, 3, 4, 5, 6]
C::[1, 2, 3, 4, 5, 6]

D::[5, 6]
c1) A>B+2
c2) B<=C*2-8
c3) D=/=C
c4) D=/=A

Si applichi l'arc-consistenza al problema così modellato, considerando i vincoli nell'ordine da c1 (arco diretto e inverso) a c4, mostrando come si riducono i domini ad ogni iterazione, fino alla quiescenza della rete.
Dire se l'applicazione dell'arc-consistenza consente, per questo CSP, di trovare una soluzione senza ricerca o meno.

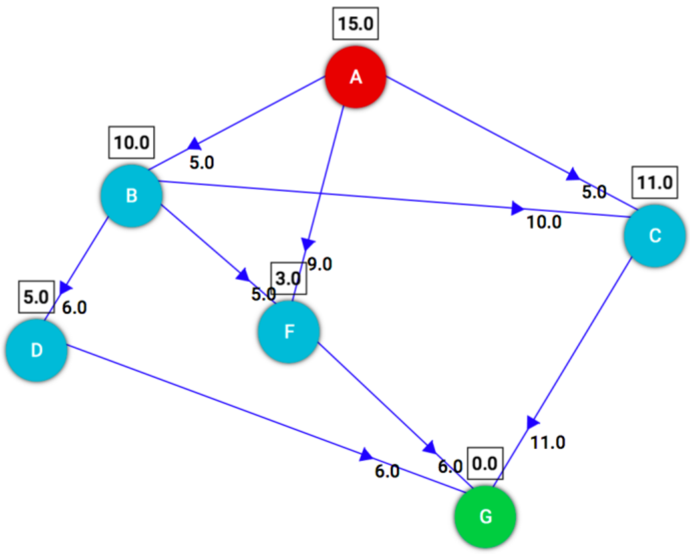
Esercizio 4 (5 punti)

Definire un programma Prolog `checkLists(List1,N)` in cui `List1` è una lista di liste di valori interi e che risulti vero se `List1` contiene tutte liste che NON contengono il valore `N`. `checkLists` risulta sempre vero se la lista `List1` è vuota. Si abbia cura di riportare la definizione di eventuali predicati ausiliari utilizzati.
Esempi:

```
?- checkLists ([[7,4],[6],[]], 3)
yes
?- checkLists ([], 3)
yes
?- checkLists ([[7,4],[2],[]], 2).
No
```

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:



a) Si applichi la ricerca **A* su alberi** (che non tiene traccia dei nodi già visitati e considera nodi diversi quelli che hanno lo stesso stato/nome ma strade diverse per raggiungerlo) e si indichino i nomi dei nodi espansi nell'ordine di espansione. Quindi nel caso si generino, da strade diverse, nodi già considerati questi verranno comunque espansi. 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.
b) Qual è il cammino e il costo di cammino trovato per arrivare al goal G a partire dal nodo iniziale A?
c) La soluzione trovata nel caso A* è ottimale? (motivare la risposta).

Esercizio 6 (4 punti)

Si introduca brevemente il metodo di ricerca a costo uniforme sottolineandone le caratteristiche. Se ne descriva poi sinteticamente l'algoritmo in pseudocodice.

11 febbraio 2021 - Soluzioni

Esercizio 1

In logica dei predicati del I ordine:

1. ∀X∀Y persona(X) ⋀ persona(Y) ⋀ bisognosa(Y) ⋀ aiuta(X,Y) -> buona(X)
2. ∀X persona(X) ⋀ not haAlloggio(X)-> bisognoso(X).
3. persona(giorgio) ⋀ not haAlloggio(giorgio).
4. persona(maria).
5. aiuta(maria,giorgio).

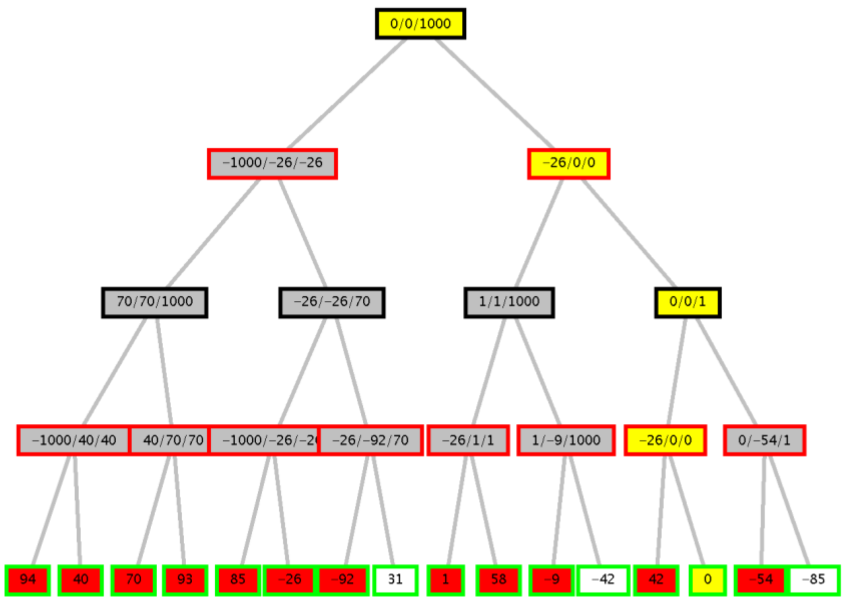
Query: buona(maria).

In forma a clausole:
C1: not aiuta(X,Y) ⋁ not persona(Y) ⋁ not persona(X) ⋁ not bisognosa(Y) ⋁ buona(X)
C2: not persona(X) ⋁ haAlloggio(X) ⋁ bisognoso(X).
C3a: persona(giorgio)
C3b: not haAlloggio(giorgio).
C4: persona(maria).
C5: aiuta(maria,giorgio).
Goal(QueryNegata): not buona(maria).

Risoluzione
Da GNeg e C1:
C6: not aiuta(maria,Y) ⋁ not persona(Y) ⋁ not persona(maria) ⋁ not bisognosa(Y).
Da C6 e C3a:
C7: not aiuta(maria,giorgio) ⋁ not persona(maria) ⋁ not bisognoso(giorgio).
Da C7 e C4:
C8: not aiuta(maria,giorgio) ⋁ not bisognoso(giorgio).
Da C8 e C2:
C9: not aiuta(maria,giorgio) ⋁ haAlloggio(giorgio) ⋁ not persona(giorgio).
Da C9 e C3b:
C10: not aiuta(maria,giorgio) ⋁ not persona(giorgio).
Da C10 e C3a:
C11: not aiuta(maria,giorgio)
Da C11 e C5:
C12: clausola vuota contraddizione!!

Esercizio 2

Min-max: strada in giallo (scelto arco a destra)– valore nodo radice 0.
Alfa-beta:



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

Archi tagliati a22, a26, a30.

Scelta per il ramo a sinistra, valore propagato 0.

Esercizio 3

A::[1, 2, 3, 4, 5, 6]
B::[1, 2, 3, 4, 5, 6]
C::[1, 2, 3, 4, 5, 6]
D::[5, 6]
c1) A>B+2
c2) B<=C*2-8
c3) D=/=C
c4) D=/=A

Risveglio c1) A{[1 .. 6]} > B{[1 .. 6]} + 2
  Rimossi [1, 2, 3] dal dominio di A{[1 .. 6]} Nuovo Dom_A{[4 .. 6]}
  Rimossi [4, 5, 6] dal dominio di B{[1 .. 6]} Nuovo Dom_B{[1 .. 3]}
Risveglio c2) B{[1 .. 3]} =< C{[1 .. 6]} * 2 - 8
  Rimossi [1, 2, 3, 4] dal dominio di C{[1 .. 6]} Nuovo Dom_C{[5, 6]}
Risveglio c3) D{[5, 6]} \= C{[5, 6]}
Risveglio c4) D{[5, 6]} \= A{[4 .. 6]}

Dom_A{[4 .. 6]}
Dom_B{[1 .. 3]}
Dom_C{[5, 6]}
Dom_D{[5, 6]}

Non è possibile trovare una soluzione senza ricerca.

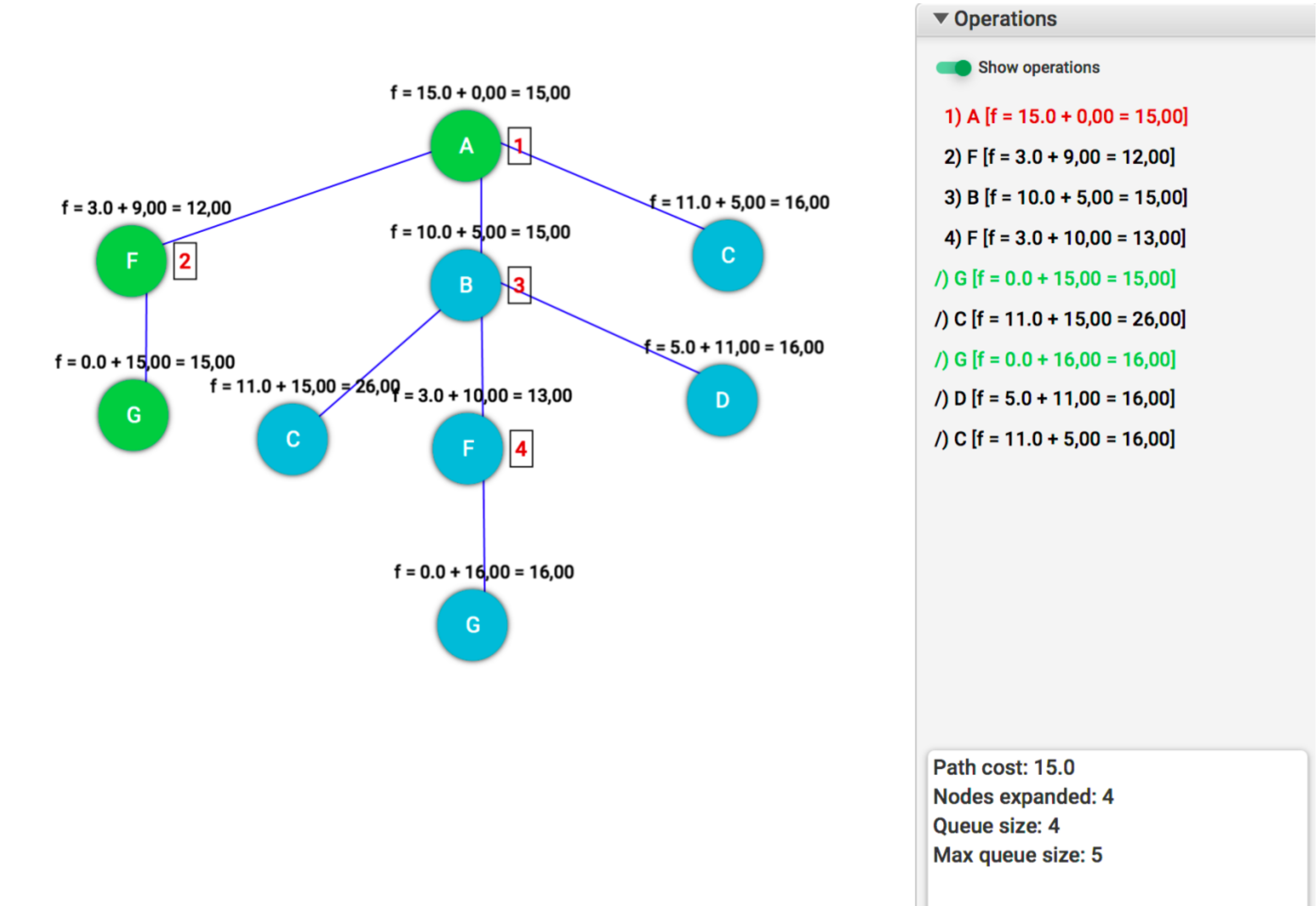
Esercizio 4

```
checkLists ([], _).
checkLists ([H|T], N):- not member(N,H), checkLists (T, N).

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

Esercizio 5

Con ricerca A*, data l'euristica che è ammissibile, si trova la soluzione ottimale:



I nodi espansi sono AFBFG. Si noti che il nodo F è considerato due volte con percorsi diversi.
Alla fine la strada selezionta è di costo 15 e è quella per cui si raggiunge il Goal G mediante il cammino migliore AFG.

Esercizio 6

Si vedano le slide del corso.