FONDAMENTI DI INTELLIGENZA ARTIFICIALE

9 Luglio 2020 – 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: RossiMario20200709AI

Esercizio 1 (6 punti)

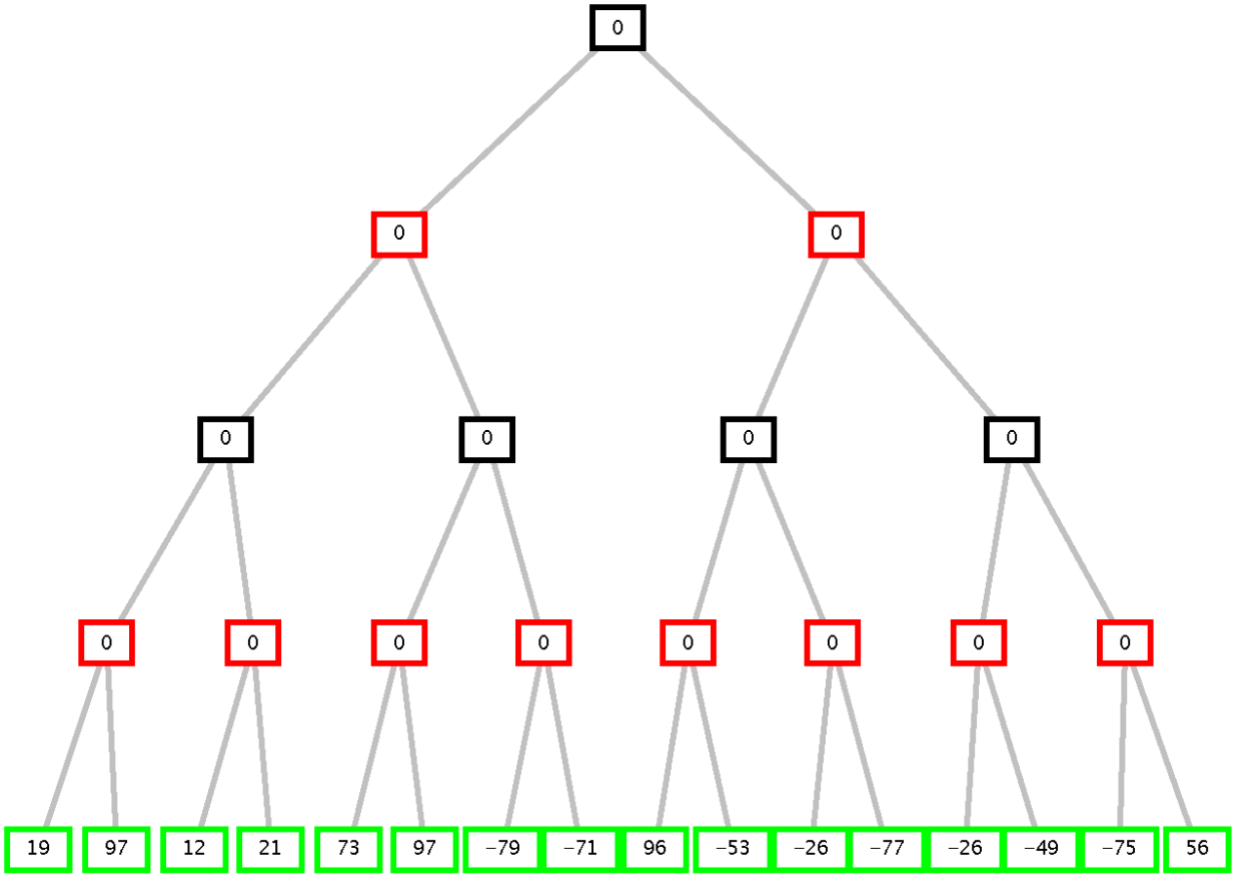
Si esprimano in logica dei predicati del I ordine le seguenti frasi:

1. Se si indossa la mascherina oppure si sta a distanza e all'aperto allora si rispetta la legge (OR NON ESCLUSIVO)
2. Mario non indossa una mascherina. Mario si trova in un parco. Mario rispetta le distanze.
3. Se ci si trova in un parco allora si è all'aperto.

Le si trasformi in clausole, e si dimostri poi, usando la risoluzione, che Mario rispetta la legge. A tal scopo si usino i seguenti predicati: rispetta_legge(X) indica che X rispetta la legge; indossa(X,Y) indica che X indossa Y; distanza(X) indica che X rispetta le distanze; si_trova(X,Y) indica che X si trova nel luogo Y; all_aperto(X) indica che X si trova all'aperto; parco(Y) indica che Y è un parco.

Esercizio 2 (5 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 19) sarà denominato a15, mentre l'ultimo arco che connette il nodo foglia più a destra (valore 56) a30.



Esercizio 3 (6 punti)

Sia dato il seguente CSP:
A::[3..5]
B,C,D::[2..6]
A<B+5
C<=B-3
A>=D+1

Mostrare i passi per arrivare alla prima soluzione, considerando le variabili con l'euristica Minimum Remaining Value (MRV), e applicando il Forward Chaining dopo il labeling di una variabile, per ridurre i domini delle variabili future. A parità di euristica, si scelga la variabile in ordine alfabetico. I valori per le variabili si scelgano a partire dal più basso.

Esercizio 4 (5 punti)

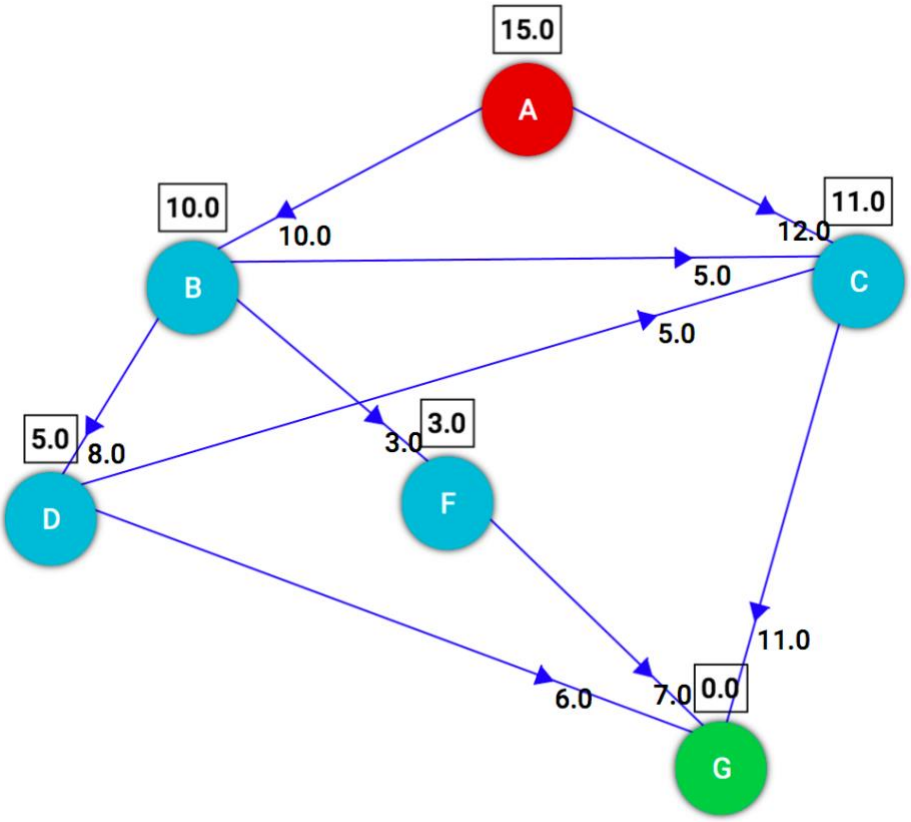
Si scriva un predicato PROLOG:
`newList(P, N, L1)`
che data una lista `P` di interi e un intero `N`, dia in uscita una nuova lista `L1` con tutti gli elementi uguali a `P` tranne l'elemento di `P` in posizione `N` che dovrà essere rimosso dalla lista `P`.
`N` è supposto minore o uguale della lunghezza della lista `P`
Se `N` è `<=0`, o `P` è vuota, `P` rimarrà invariata.
Ad esempio:

```
?- newList([3,2,3,1], 2, X).
X = [3, 3, 1]

?- newList([3,2,3,1], 0, X).
X = [3, 2, 3, 1]
```

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:



a) Si applichi la ricerca **greedy best first** e si indichino i nodi espansi nell'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.
b) Qual è il costo di cammino trovato per arrivare al goal G a partire dal nodo iniziale A?
c) La soluzione trovata in questo caso è ottimale? (motivare la risposta).
d) In generale, la ricerca best-first garantisce o non garantisce l'ottimalità? (motivare la risposta).

Esercizio 6 (4 punti)

Dopo avere spiegato brevemente il predicato predefinito Prolog: `call(X)` si consideri il seguente predicato Prolog:

```
p(3).
c(1,Z) :-p(Z).
```

E si indichino le risposte Prolog alle seguenti due query:

```
?- call(p(X)).
?- call(c(Y,3)).
```

9 Luglio 2020 - Soluzioni

Esercizio 1

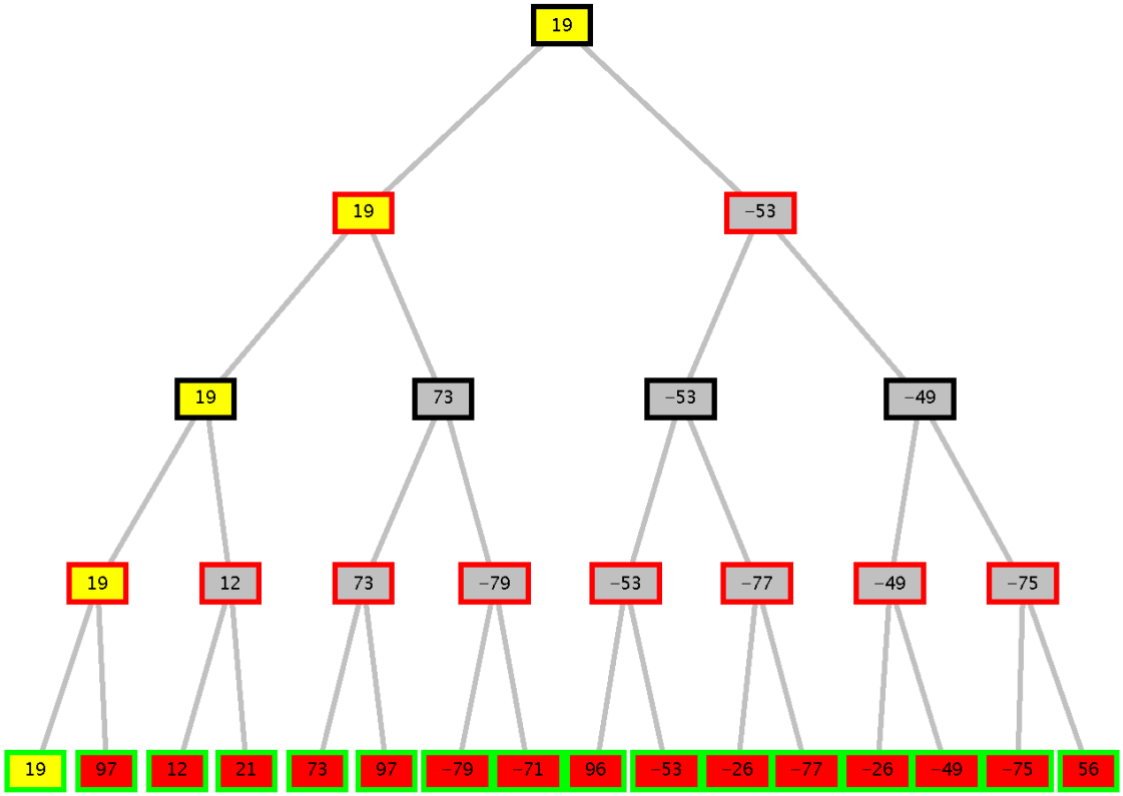
1.: ∀X ∀Y indossa(X,mascherina) ∨ (distanza(X) ∧ all'aperto(X))→rispetta_legge(X).
2.: ¬ indossa(mario,mascherina) ∧ ∃Y (si_trova(mario,Y) ∧ parco(Y)) ∧ distanza (mario).
3.: ∀X ∀Y si_trova(X,Y) ∧ parco(Y)→ all_aperto(X).
Goal negato: ¬rispetta_legge(mario).

Tradotti in clausole:
1a.: ¬ indossa(X,mascherina)∨ rispetta_legge(X).
1b.: ¬distanza(X) ∨ ¬all'aperto(X) ∨ rispetta_legge(X).
2a.: ¬ indossa(mario,mascherina).
2b.: si_trova(mario,c1). skolem
2c.: parco(c1)
2d.: distanza (mario).
3.: ¬si_trova(X,Y) ∨ ¬parco(Y) ∨ all_aperto(X).
4. GNeg.: ¬rispetta_legge(mario).

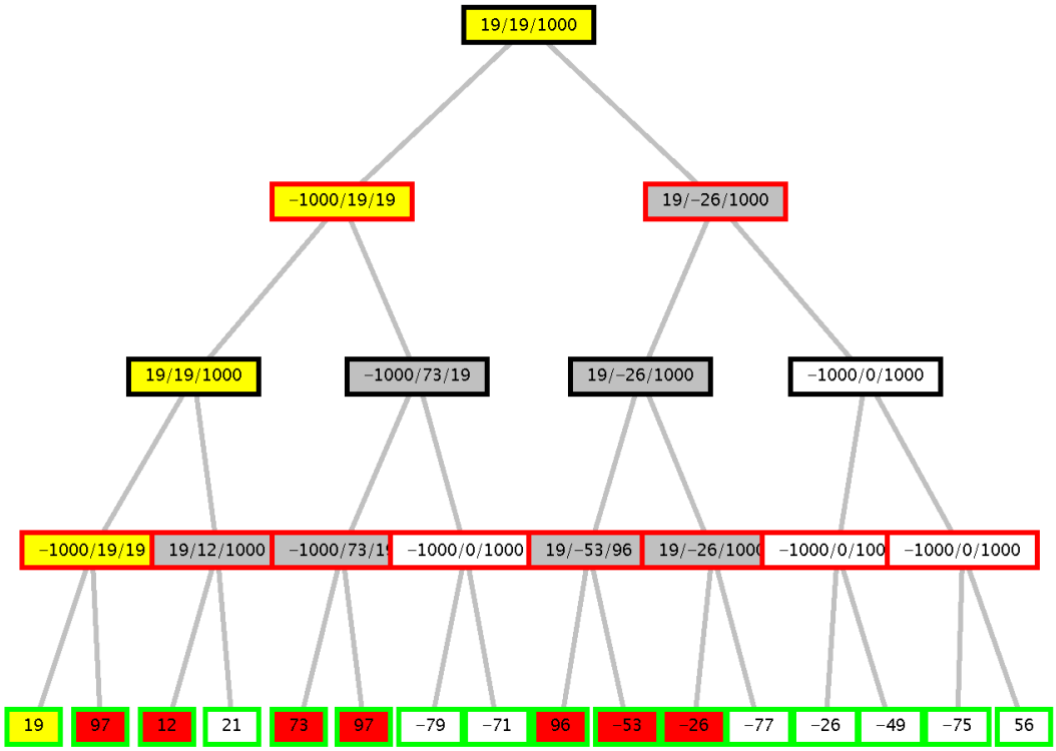
Refutazione:
5.: GNeg+1b. : ¬distanza(mario) ∨ ¬all_aperto(mario)
6.: 5+2d.: ¬all'aperto(mario)
7.: 6 + 3: ¬si_trova(mario,Y) ∨ ¬parco(Y)
8: 7 + 2c: ¬si_trova(mario,c1)
9: 8 + 2b: contraddizione logica!

Esercizio 2

Min-max:



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: a18,a10,a26, a6. Max seleziona la mossa a sinistra. Valore propagato 19.

Esercizio 3

A::[3, 4, 5]				
B::[2, 3, 4, 5, 6]				
C::[2, 3, 4, 5, 6]				
D::[2, 3, 4, 5, 6]				
C<=B-3				
A>=D+1				
	A	B	C	D
Labeling di A	A=3	[2...6]	[2...6]	[2...6]
FC	A=3	[2...6]	[2...6]	[2]
Labeling di D	A=3	[2...6]	[2...6]	D=2
FC	A=3	[2..6]	[2..6]	D=2
Labeling di B	A=3	B=2	[2...6]	D=2
FC	A=3	B=2	Fail, quindi backtracking	D=2
Labeling di B	A=3	B=3	[2...6]	D=2
FC	A=3	B=3	Fail, backtracking	D=2
Labeling di B	A=3	B=4	[2...6]	D=2
FC	A=3	B=4	Fail, backtracking	D=2
Labeling di B	A=3	B=5	[2...6]	D=2
FC	A=3	B=5	[2]	D=2
Labeling di C	A=3	B=5	C=2	D=2

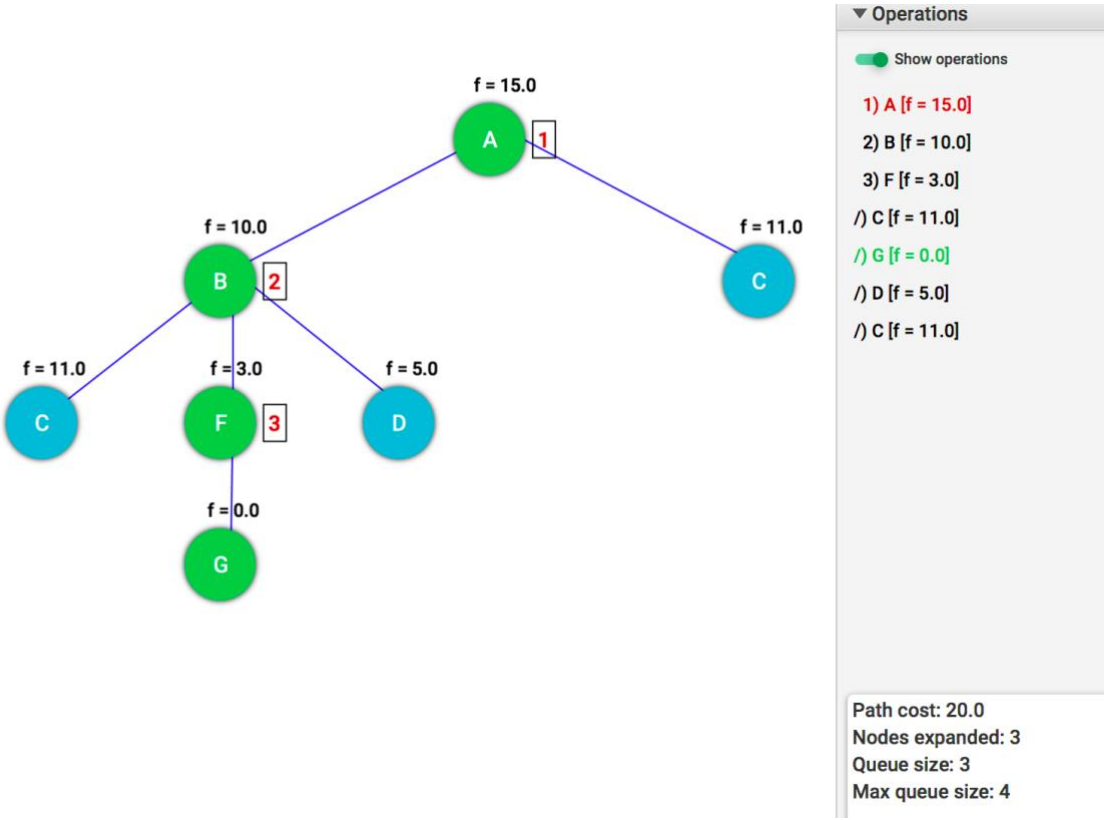
Esercizio 4

```
newList([], _ , []):-!.
newList(P, N, P) :- N =< 0,!.
newList([_|Tail], 1, Tail) :- !.
newList([H|Tail], N, [H|L]) :- N1 is N-1, newList(Tail, N1, L).
```

Oppure:

```
newList([], _, []).
newList(L, N, L):- N =< 0, !.
newList(L, N, X):- newList(L, N, X, 1).
newList([_|T], N, T, N).
newList([H|T], N, [H|T1], N1):- N2 is N1 + 1, newList(T, N, T1, N2).
```

Esercizio 5



Il costo di cammino trovato è 20 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. In questo caso viene trovata la soluzione ottimale (20), ma in generale la ricerca best-first non garantisce l'ottimalità considerando ad ogni passo di espansione per la selezione del prossimo nodo solo la stima della sua distanza dal goal, ma non il costo del cammino percorso per arrivarci.

Esercizio 6

Per il predicato Prolog `call` si veda il materiale del corso.
Per le due query nell'esercizio le risposte sono le seguenti:

```
?- call(p(X)).   Yes X =3
?- call(c(Y,3)).    Yes Y=1
```