FONDAMENTI DI INTELLIGENZA ARTIFICIALE

11 Luglio 2019 – Tempo a disposizione: 2 h – Risultato: 32/32 punti

Esercizio 1 (punti 7)

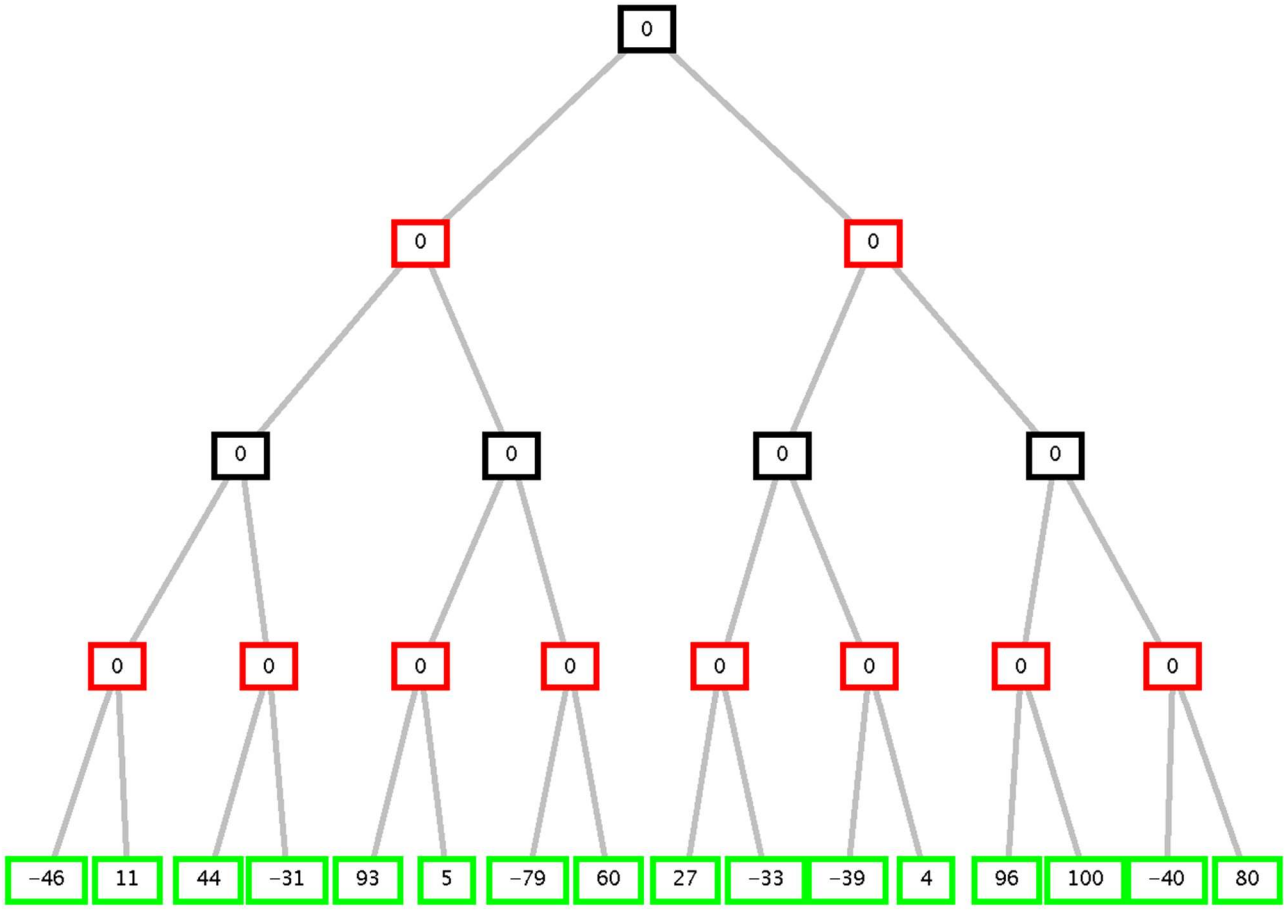
Scrivere le seguenti frasi in logica dei predicati del primo ordine:

1. *Ogni cucciolo è di colore bianco o di colore nero (OR esclusivo)*
2. *Tutti i cuccioli neri sono maschi*
3. *Tutti i cuccioli sono maschi o femmine (OR esclusivo)*
4. *Bob è un maschio e Rose è una femmina ed entrambi sono cuccioli.*

Trasformarle poi in clausole e dimostrare tramite risoluzione che *esiste almeno un cucciolo di colore bianco.* A tal scopo, si usino i seguenti predicati (dal significato inteso): **`cucciolo(X)`**: X è un cucciolo; **`colore(X, Y)`**: il colore di X è Y; **`maschio(X)`**: X è un maschio; **`femmina(X)`**: X è una femmina.

Esercizio 2 (punti 4)

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 (punti 5)

Dato il seguente programma Prolog:

```
subset([],[]).
subset([X|T],[X|T1]):- !,subset(T,T1).
subset([_|T],T1):- subset(T,T1).
```

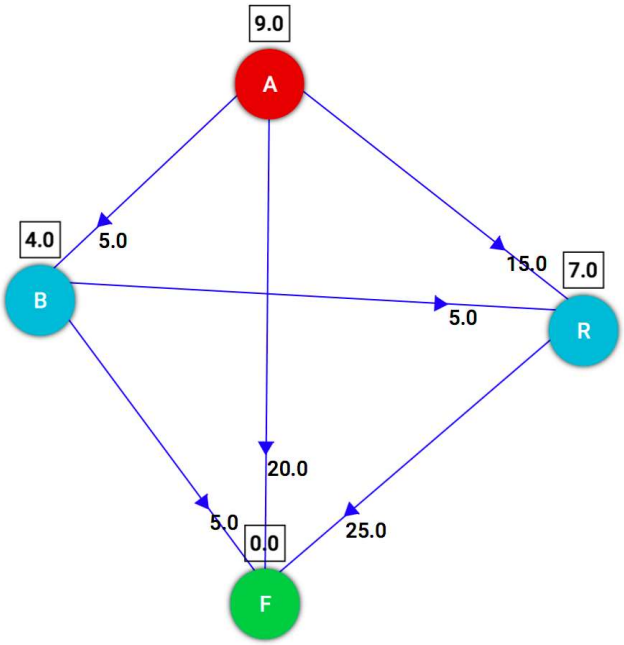
disegnare l'albero SLD per il goal seguente (si indichino i tagli effettuati dal *cut* e non si espandano i rami tagliati):

```
?-subset([9,11,16],[9,16]).
```

Esercizio 4 (punti 6)
Definire un programma Prolog `eachLists(List1,Min,Max)` in cui `List1` è una lista di liste di valori interi e che risulti vero se `List1` contiene tutte liste che rispettano questa proprietà:
I valori dei loro elementi sono compresi fra `Min` e `Max` (`Min` e `Max` inclusi, dove `Min<=Max`). La proprietà è sempre garantita se la lista è vuota.
`eachLists` risulta sempre vero se la lista `List1` è vuota.
Esempi:

```
?- eachLists([[7,4],[6],[]], 3, 8)
yes
?- eachLists([], 3, 8)
yes
?- eachLists([[7,4],[2],[]], 3, 8).
No
```

Esercizio 5 (punti 5)
Si consideri il seguente grafo, dove A è il nodo iniziale e F 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. A fianco di ogni nodo, in un quadrato, è indicata inoltre la stima euristica della sua distanza dal nodo goal:



a) Si applichino prima la ricerca Best-first e successivamente la ricerca A* e si disegnino gli alberi di ricerca sviluppati 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. Si consideri come euristica h(*n*) quella indicata nel quadrato a fianco di ogni nodo in figura.
b) L'euristica è ammissibile?
c) Qual è il costo di cammino trovato da Best-first e quale quello trovato da A*?

Esercizio 6 (punti 5)
Si consideri il problema delle 4 regine (inserire 4 regine su una scacchiera 4x4 in modo che non si attacchino). Si ricorda che le regine attaccano in orizzontale, verticale e diagonali.
Si rappresenti il problema come CSP modellando ogni variabile/regina con una colonna e considerando i valori del dominio dati dalle possibili righe (1,…,4) in cui posizionare le regine. Si rappresentino variabili, domini e i vincoli in cui sono coinvolte.

Si risolva poi il problema applicando la tecnica del forward checking. Si considerino le variabili dando priorità a quelle che rappresentano colonne con indice più basso. Per la scelta dei valori del dominio si considerino prioritari quelli con valore minore. Si evidenzino i domini e come si modificano ad ogni passo, assieme ai punti di backtraking (che si svolgerà in modo cronologico).

Si risolva poi nuovamente il problema applicando il Partial Look Ahead (PLA) ad ogni passo dopo il forward checking. Cosa cambia relativamente al backtraking?

11 Luglio 2019 - Soluzioni

Esercizio 1

Traduzione in predicati in logica del primo ordine:

Ogni cucciolo è di colore bianco o di colore nero (OR esclusivo)

1. ∀X cucciolo(X)→ colore(X,bianco) xor colore(X,nero)

Tutti i cuccioli neri sono maschi.

2. ∀X cucciolo(X) and colore(X, nero) → maschio(X)

Tutti i cuccioli sono o maschi o femmine (xor).

3. ∀X cucciolo(X) → maschio(X) xor femmina(X)

Bob è un maschio e Rose è una femmina, entrambi sono cuccioli.

4. maschio(bob)
5. cucciolo(bob)
6. femmina(rose)
7. cucciolo(rose)

Goal ∃X cucciolo(X) and colore(X, bianco)

Negazione del Goal: not(∃X cucciolo(X) and colore(X, bianco))
Equivalente a: ∀X (not cucciolo(X) or not colore(X, bianco)

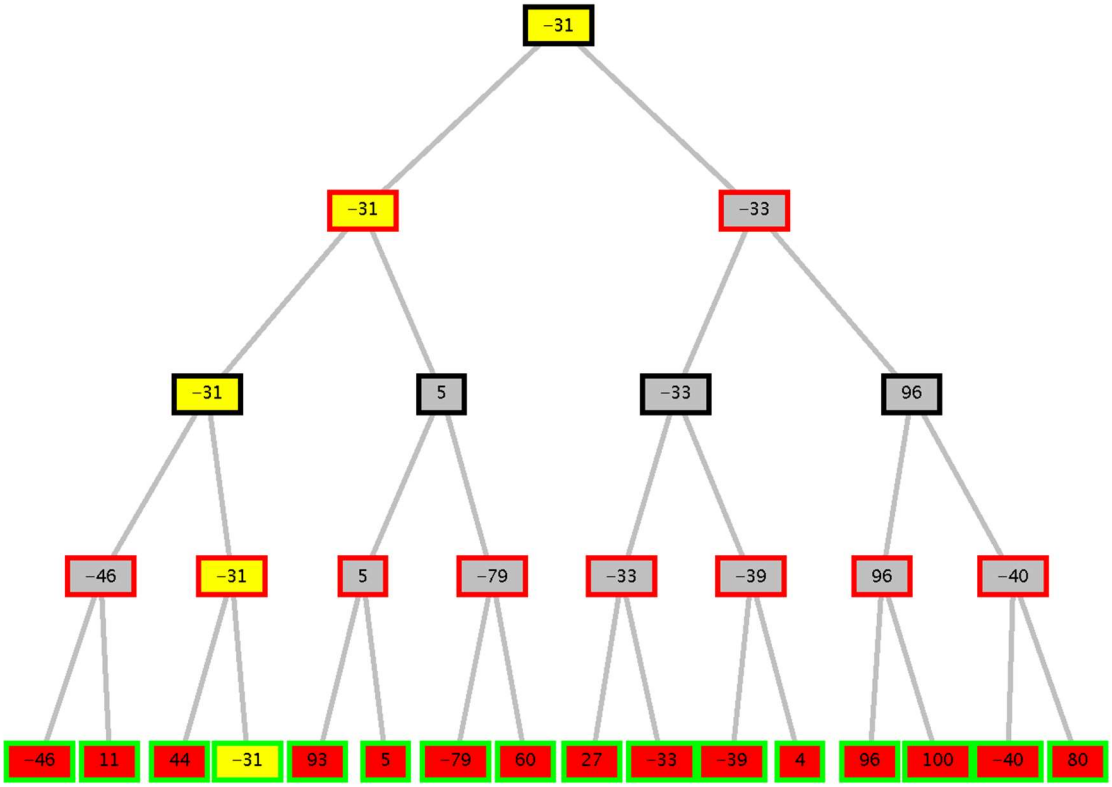
Traduzione in clausole:

```
1a.   not cucciolo(X) or colore(X,bianco) or colore(X,nero)
1b.   not cucciolo(X) or not colore(X,bianco) or not colore(X,nero)
2.    not cucciolo (X) or not colore (X, nero) or maschio(X)
3a.   not cucciolo (X) or maschio(X) or femmina(X)
3b.   not cucciolo (X) or not maschio(X) or not femmina(X)
4.    maschio(bob)
5.    cucciolo(bob)
6.    femmina(rose)
7.    cucciolo(rose)
GNeg.:     not cucciolo(X) or not colore(X, bianco)
```

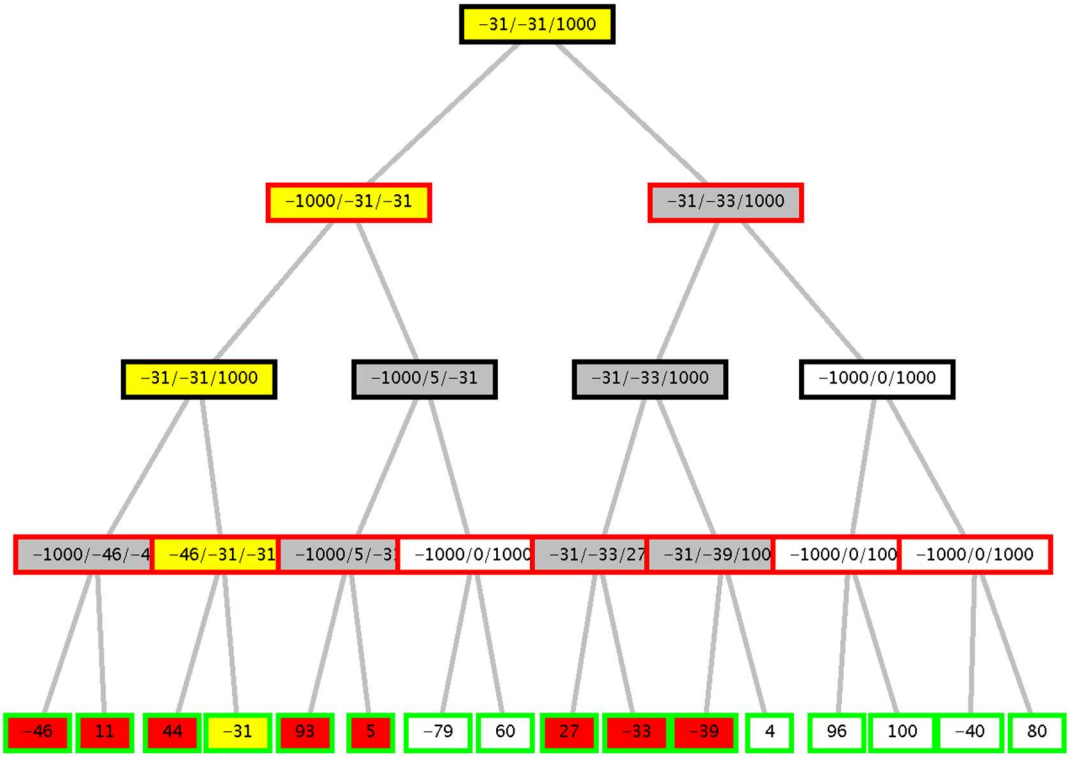
Risoluzione:

8: GNeg + 1a: not cucciolo(X) or colore(X,nero)
9: 8+7: colore(rose,nero)
10: 9+2: not cucciolo(rose) or maschio(rose)
11: 10+3b: not cucciolo(rose) or not femmina(rose)
12: 11 + 7: not femmina(rose)
13: 12 + 5: Contraddizione!!

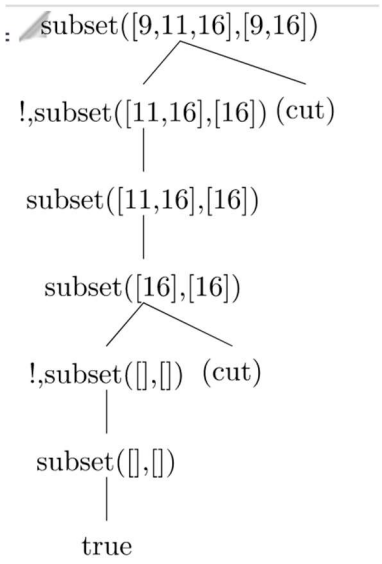
Esercizio 2
Min-Max:



Tagli alfa-beta: Sono 3 Tagli



Esercizio 3

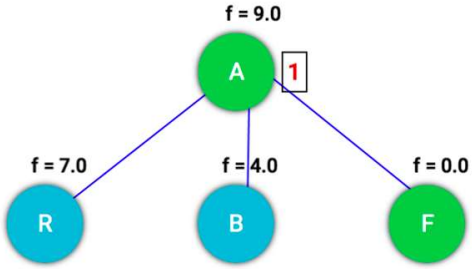


Esercizio 4

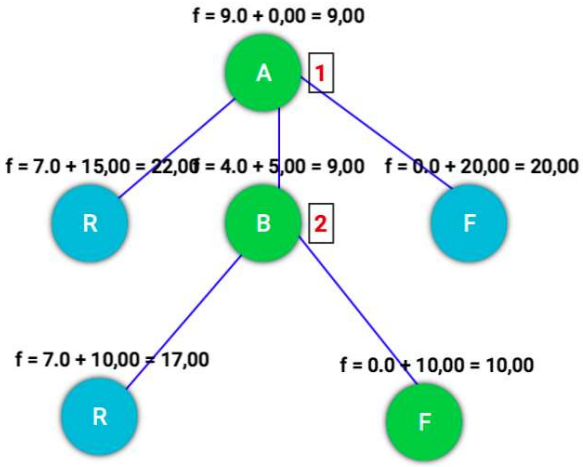
```
%%definizione del predicato rangeList
rangeList([], Min, Max).
rangeList([A|B], Min, Max) :- A =< Max, A >= Min, rangeList(B, Min, Max).

eachLists([], Min, Max).
eachLists([H|T], Min, Max):- rangeList(X,Min, Max), eachLists(T, Min, Max).
```

Esercizio 5

Best-first, costo di cammino 20:



A*, costo di cammino 10:



L'euristica è ammissibile.

Esercizio 6

Modellazione
Variabili: X1, X2, X3, X4
Valori dei Domini: 1,2,3,4
Vincoli X1 =/= X2 =/=X3=/=X4
|Xi-Xj| =/= |i-j|

Solo Forward Checking (2 backtraking effettuati)

X1	X2	X3	X4	
[1..4]	[1..4]	[1..4]	[1..4]	
1	[3,4]	[2,4]	[2,3]	
1	3	[]	[2]	fallimento dominio X3 vuoto; backtraking cronologico
1	4	[2]	[3]	
1	4	2	[]	fallimento dominio X4 vuoto; backtraking cronologico
2	[4]	[1,3]	[1,3,4]	
2	4	[1]	[1,3]	
2	4	1	[3]	
2	4	1	3	

PLA: (1 solo backtraking svolto)

X1	X2	X3	X4	
[1..4]	[1..4]	[1..4]	[1..4]	
1	[3,4]	[2,4]	[2,3]	dopo Forward checking
1	[4]	[4]	[2,3]	dopo PLA
1	4	[]	[3]	dopo Forward checking; backtraking cronologico
2	[4]	[1,3]	[1,3,4]	dopo forward checking
2	[4]	[1,3]	[1,3,4]	dopo PLA
2	4	[1]	[1,3]	dopo FC
2	4	[1]	[1,3]	dopo PLA
2	4	1	[3]	dopo FC (e PLA)
2	4	1	3	