FONDAMENTI DI INTELLIGENZA ARTIFICIALE

12 Settembre 2019 – Tempo a disposizione: 2 h – Risultato: 32/32 punti

Esercizio 1 (6 punti)

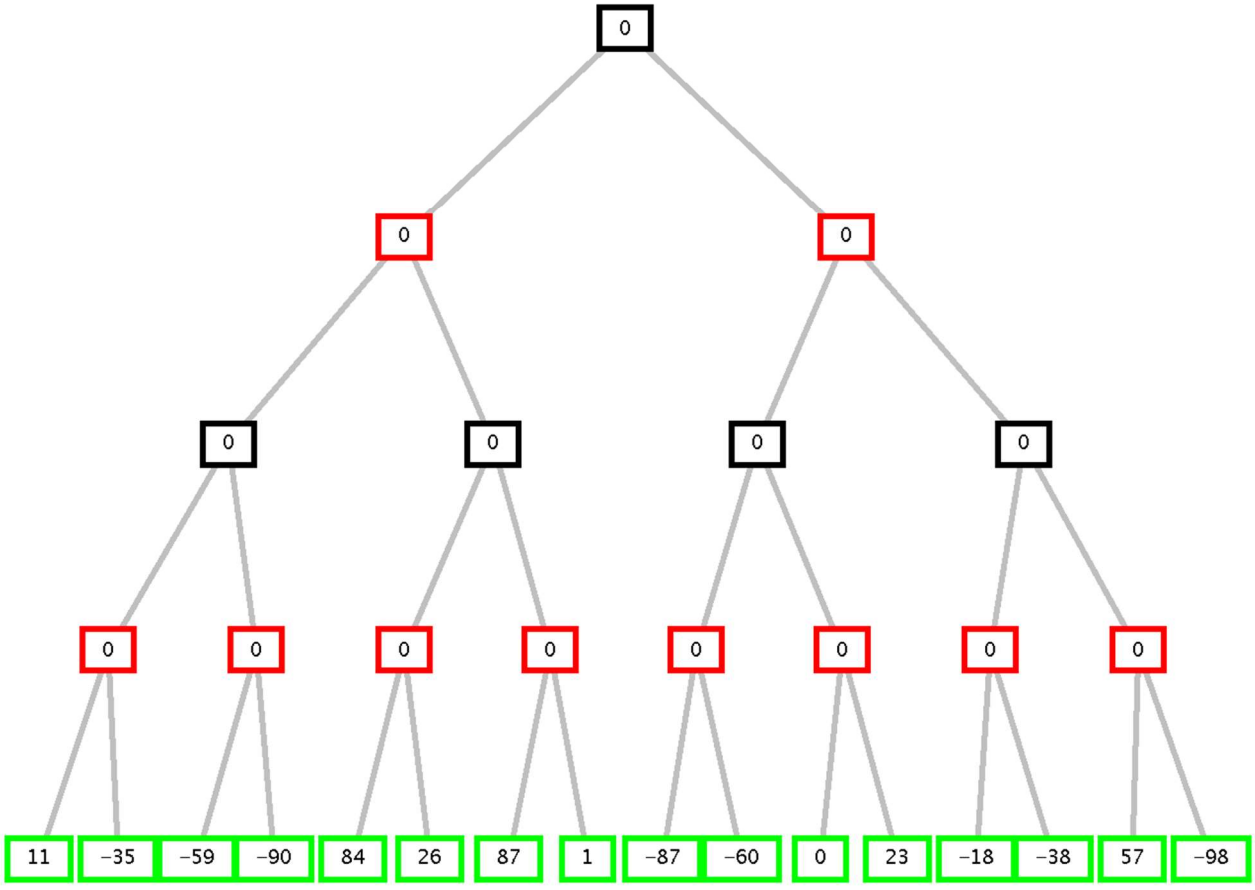
Si formalizzino le seguenti frasi in logica dei predicati:

- Esiste almeno un gatto amichevole
- Ogni gatto amichevole ama i bambini
- Ogni gatto che ama i bambini fa le fusa
- Ogni gatto che fa le fusa ama i bambini
- Il gatto Attila è amichevole.

Le si trasformi in clausole e si usi poi il principio di risoluzione per dimostrare che c'è un gatto che fa le fusa. Si usino i predicati gatto(X), amichevole(X), ama_bambini(X), e il predicato fa_fusa(X).

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 6)

Dato il seguente programma Prolog, `rangeList(L1, L2, Min, Max)` che riporta nella lista `L2` gli elementi della lista `L1` compresi nel range tra `Min` e `Max`:

```
rangeList([], [], _, _).
rangeList([A|B], [A|T], Min, Max) :-
                    A =< Max, A >= Min, !, rangeList(B,T,Min,Max).
rangeList([_|B], T, Min, Max) :- rangeList(B,T,Min,Max).
```

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

```
?-rangeList([5,12,31], L, 1, 10).
```

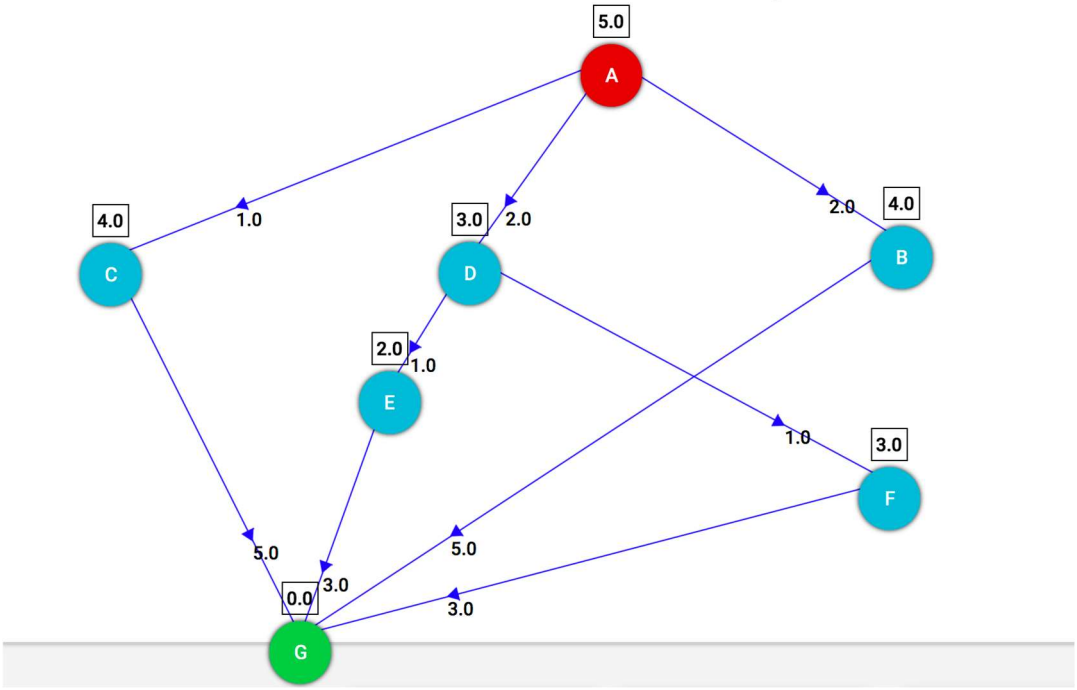
Esercizio 4 (punti 4)

Data una lista di `reali` di nome `L` che contiene un numero pari di elementi, si realizzi un predicato Prolog `extract(L,L1)` che restituisca una nuova lista `L1` contenente solo quegli elementi della lista `L` in modo che, considerati a coppie, a partire dai primi due elementi, il primo elemento della coppia sia minore o uguale (ovvero preceda) il secondo elemento.

Ad esempio, se `L = [3.2, 4.9, -1.1,-2.7, 5.3, 0]`, il predicato `extract` restituisce `[3.2, 4.9]`, in quanto `3.2` precede `4.9`, mentre non contiene gli altri elementi in quanto `-1.1` non precede `-2.7` e `5.3` non precede `0`.

Esercizio 5 (punti 7)

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. A fianco di ogni nodo, in un quadrato, è indicata inoltre la stima euristica della sua distanza dal nodo goal:



a) L'euristica è ammissibile?
b) Si applichi la ricerca A* e si disegni l'albero di ricerca sviluppato 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 del loro nome. Si consideri come euristica h(*n*) quella indicata nel quadrato a fianco di ogni nodo in figura.
c) Qual è il costo di cammino trovato da A* ed il numero di nodi espansi per arrivare al goal G a partire dal nodo iniziale A?

Esercizio 6 (punti 5)

Si consideri il seguente CSP:
A::[4, 5, 6, 7, 8, 9, 10]
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-5
C>B-7
A=D+3
B=D+2

Si cerchi la prima soluzione, applicando labeling e Forward Checking dopo ogni passo di labeling, considerando le variabili secondo l'ordine alfabetico del loro nome, e i valori nel dominio secondo l'ordine crescente sugli interi.

Si cerchi poi sempre la prima soluzione, ma applicando l'euristica Minimum Remaining Value per la scelta della variabile, e sempre applicando labeling e Forward Checking dopo ogni passo di labeling. Cosa cambia? Commentare adeguatamente.

COGNOME NOME_____
Min-max

Alfa-beta

12 Settembre 2019 - Soluzioni

Esercizio 1

- Esiste almeno un gatto amichevole
  $\exists X(\text{gatto}(X) \wedge \text{amichevole}(X))$
- Ogni gatto amichevole ama i bambini
  $\forall X(\text{gatto}(X) \wedge \text{amichevole}(X) \rightarrow \text{ama\_bambini}(X))$
- Ogni gatto che ama i bambini fa le fusa
  $\forall X (\text{ama\_bambini}(X) \wedge \text{gatto}(X) \rightarrow \text{fa\_fusa}(X))$
- Ogni gatto che fa le fusa ama i bambini
  $\forall X(\text{gatto}(X) \wedge \text{fa\_fusa}(X) \rightarrow \text{ama\_bambini}(X))$
- Il gatto Attila è amichevole.
  $\text{gatto}(\text{attila}) \wedge \text{amichevole}(\text{attila})$

C'è un gatto che fa le fusa.
Query: $\exists Y\ \text{gatto}(Y) \wedge \text{fa\_fusa}(Y)$

Clausole:

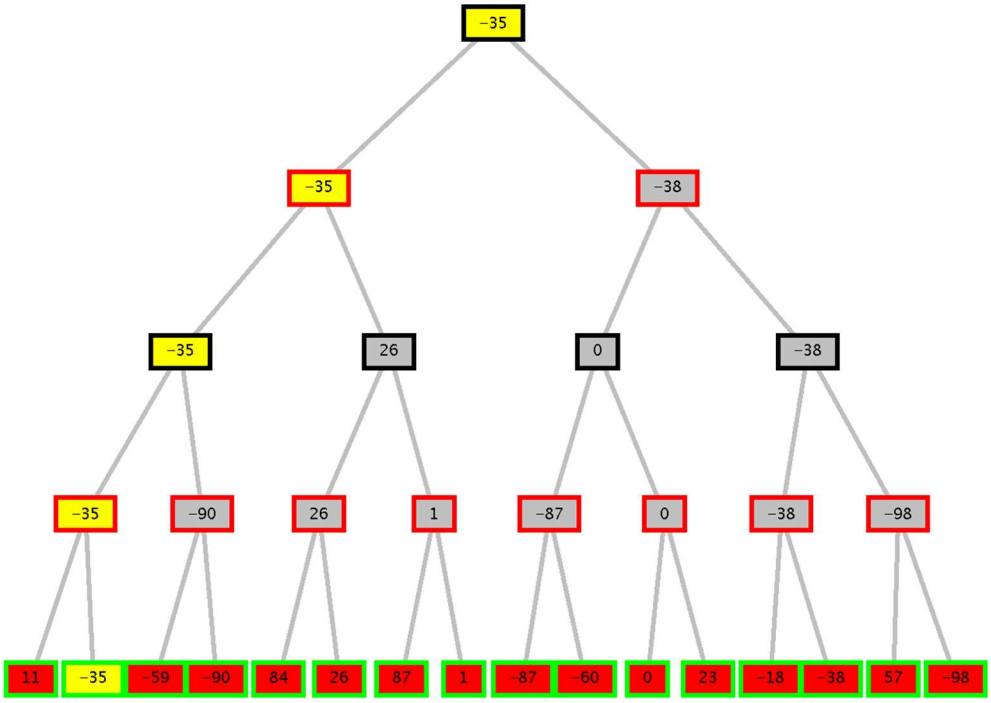
- C1a gatto(c)
- C1b amichevole(c)
- C2 $\neg$gatto(X) $\vee$ $\neg$amichevole(X) $\vee$ ama_bambini(X)
- C3 $\neg$gatto(X) $\vee$ $\neg$ ama_bambini(X) $\vee$ fa_fusa(X)
- C4 $\neg$gatto(X) $\vee$ ama_bambini(X) $\vee \neg$ fa_fusa(X)
- C5a gatto(attila)
- C5b amichevole(attila)
- C6 $\neg$gatto(Y) $\vee$ $\neg$fa_fusa(Y) (goal, query negata)

Risoluzione (una possibile "strada"):

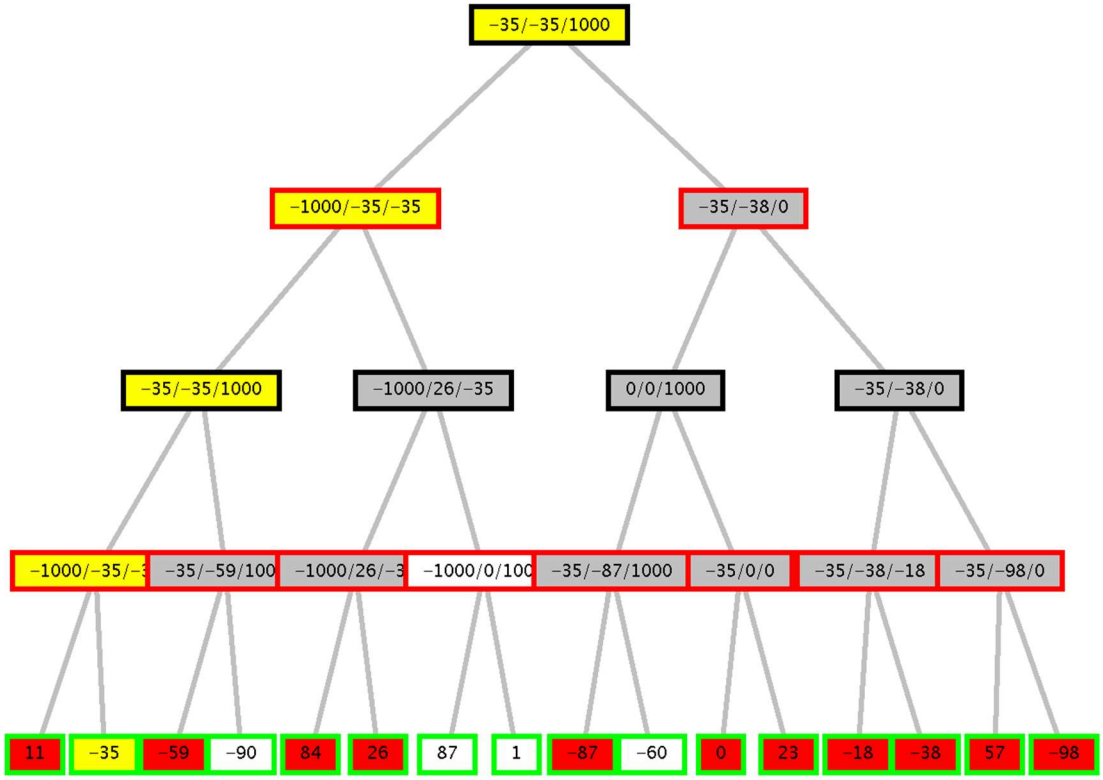
C7: C6 + C3: $\neg$gatto(X) $\vee$ $\neg$ ama_bambini(X)
C8: C7 + C2: $\neg$gatto(X) $\vee$ $\neg$amichevole(X)
C9: C8 + C5b: $\neg$gatto(attila)
C10: C9 + C5a: contraddizione

Esercizio 2

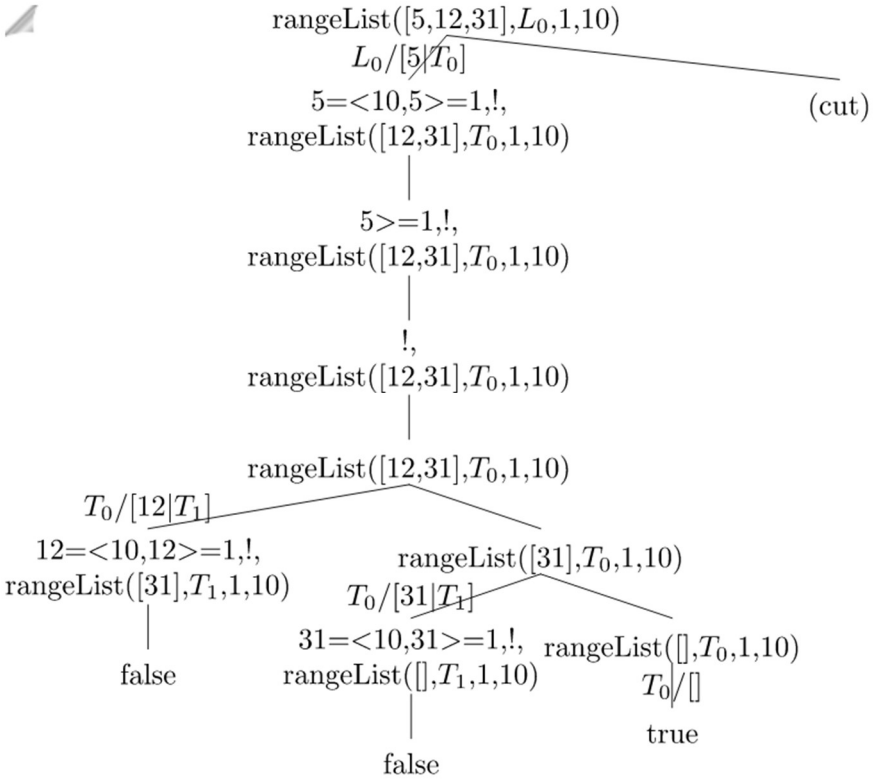
Min-Max:



Tagli alfa-beta: Sono 3 Tagli



Esercizio 3

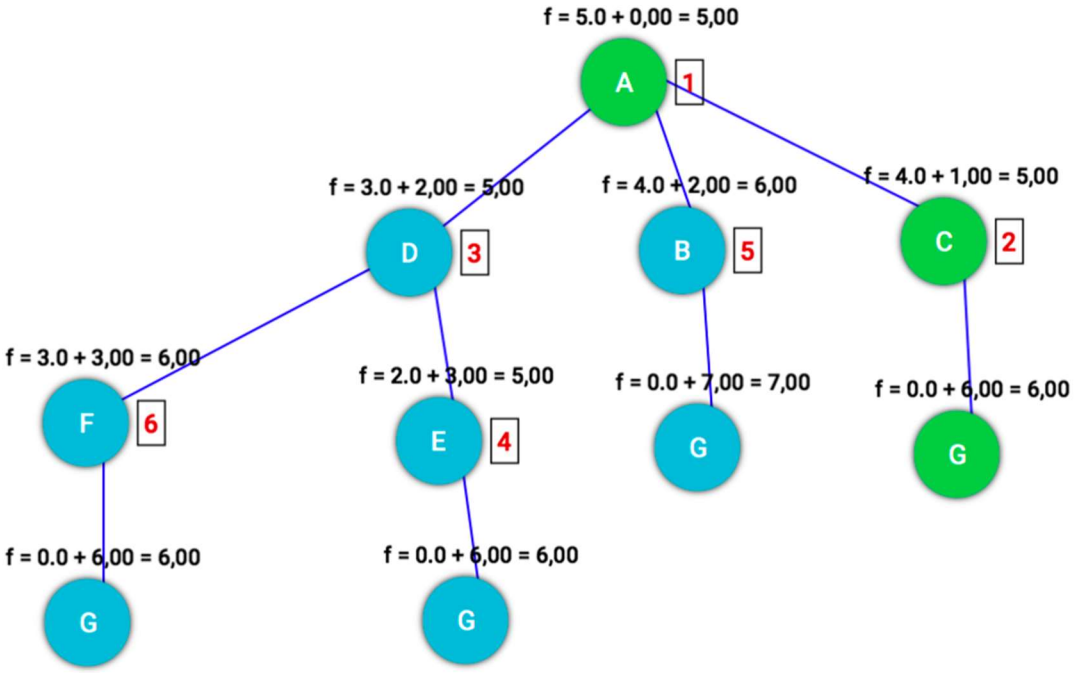


Esercizio 4

```
extract([], []).
extract([A,B|Tail],[A,B|Rest]) :- A=< B, !, extract(Tail,Rest).
extract([_ ,_|Tail],Rest):- extract(Tail,Rest).
```

Esercizio 5

A* (l'euristica è ammissibile; costo di cammino: 6; nodi espansi: 6, quelli con etichetta quadrata a fianco, con all'interno numero d'ordine di espansione)



Esercizio 6

A::[4, 5, 6, 7, 8, 9, 10]
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-5
C>B-7
A=D+3
B=D+2

	A	**B**	**C**	**D**
Labeling e FC	A=4	[1...8]	[1...10]	[1...10]
Labeling e FC	A=4	[1...8]	[1...10]	[1]
Backtracking	A=4	B=1	[1...10]	***Fail***
Backtracking	A=4	B=2	[1...10]	***Fail***
Labeling e FC	A=4	B=3	[1...10]	[1]
Labeling e FC	A=4	B=3	C=1	[1]
Labeling e FC	A=4	B=3	C=1	D=1
Con MRV invece:	**A**	**B**	**C**	**D**
Labeling e FC	A=4	[1...8]	[1...10]	[1]
Labeling e FC	A=4	[1...8]	[1...10]	D=1
Labeling e FC	A=4	[3]	[1...10]	D=1
Labeling e FC	A=4	B=3	[1...10]	D=1
Labeling e FC	A=4	B=3	C=1	D=1

L'euristica MRV fa scegliere prima la variabile con dominio più piccolo come cardinalità, e quindi porta a legare prima la variabile D all'unico valore possibile (ovvero il valore 1), evitando i fallimenti che si hanno nel caso di scelta della variabile B da istanziare prima della variabile D.