

## FONDAMENTI DI INTELLIGENZA ARTIFICIALE

9 Settembre 2022 – Tempo a disposizione: 2 h – Risultato: 32/32 punti

### Esercizio 1 (6 punti)

Si traducano le seguenti frasi nella logica dei predicati del primo ordine, poi in forma a clausole:

- Maria ama i cibi italiani.
- Chi ama i cibi italiani è goloso di pasta o pizza (non esclusivo).
- Chi è goloso di pasta non ama le zuppe.
- Chi è goloso di pizza ama il pomodoro.
- Maria non ama ciò che ama Giulia.
- Giulia ama il pomodoro.
- Tutti sono golosi di qualcosa.

Si usino i seguenti predicati:

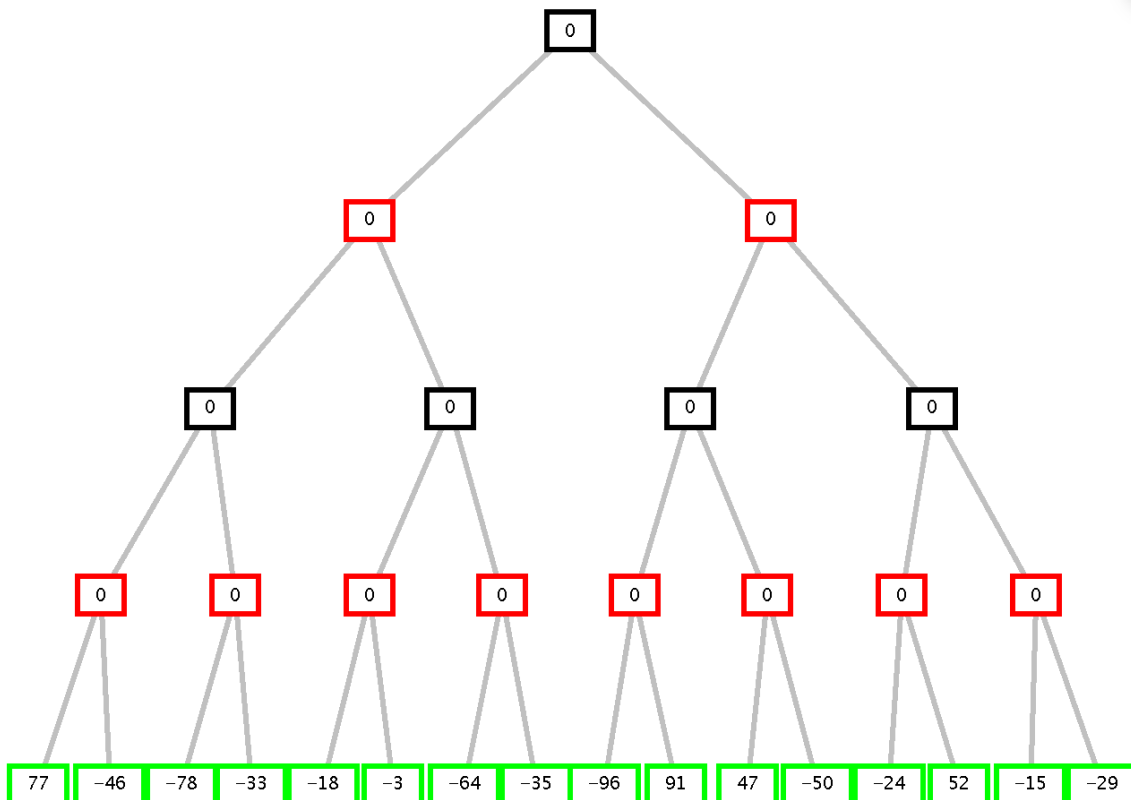
- **ama(X, Y)**: X ama Y
- **goloso(X, Y)**: X è goloso di Y

e le costanti: **pizza, pasta, zuppe, cibitaliani, pomodoro, giulia, maria**.

Si applichi poi il principio di risoluzione per dimostrare che esiste qualcuno goloso di pasta che non è goloso anche di pizza.

### 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 (6 punti)

Si scriva in Prolog il predicato **dellList(L1,L2,L3)**, che risulta vero quando **L3** rappresenta la lista degli elementi di **L2** in cui sono stati eliminati tutti gli elementi che compaiono in **L1**. A tal fine si utilizzi opportunamente (riportandolo anche nella soluzione) il predicato delete presentato anche a lezione.

Esempi

```
?- dellList([1,2,3], [1,2,3,4,2,1,5], X)
```

```
X = [4, 5]
```

```
?-dellList([8], [1,2,3,4,2,1,5], X)
```

```
X = [1, 2, 3, 4, 2, 1, 5]
```

```
?-delList([], [1,2,3,4,2,1,5],X)
X = [1, 2, 3, 4, 2, 1, 5]
?-delList([1,7,3], [],X)
X = []
```

#### Esercizio 4 (5 punti)

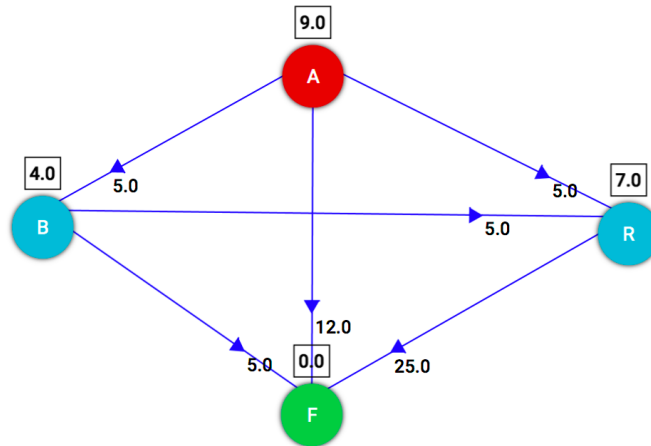
Il seguente programma Prolog restituisce una lista ordinata di termini che è la fusione delle due liste ordinate di termini date.

```
mergesort(X, [], X).
mergesort([], Y, Y).
mergesort([X|TX], [Y|TY], [X|T]) :- X<Y, !, mergesort(TX, [Y|TY], T).
mergesort([X|TX], [Y|TY], [Y|T]) :- mergesort([X|TX], TY, T).
```

Si mostri l'albero SLD relativo al goal: `?- mergesort([3,7], [4], X).`

#### Esercizio 5 (6 punti)

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. Vicino ad ogni nodo, in un quadrato, è indicata inoltre la stima euristica della sua distanza dal nodo goal F:



Si applichi la ricerca **A\*** su alberi (non tenendo quindi traccia dei nodi già visitati) **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;
- se è garantita o meno l'ottimalità.

Si mostri poi cosa succederebbe invece con ricerca **Greedy-Best-First**.

#### Esercizio 6 (4 punti)

Dopo avere brevemente introdotto le tecniche di propagazione di Forward Cheking (FC) e Partial (PLA) e Full (FLA) Look Ahead se ne mostri l'esecuzione su questo esempio mostrando la riduzione dei domini delle restanti variabili quando viene istanziata la variabile A a 3 (considerare le variabili secondo l'ordine alfabetico). PLA e FLA si facciano partire dopo avere applicato il FC.

A,B,C ::[1,2,3,4,5,6,7]

A>B

B>C

A>C

A=3

## 9 Settembre 2022 - Soluzioni

### Esercizio 1

- Maria ama i cibi italiani.
- Chi ama i cibi italiani è goloso di pasta o pizza (non esclusivo).
- Chi è goloso di pasta non ama le zuppe.
- Chi è goloso di pizza ama il pomodoro.
- Maria non ama ciò che ama Giulia.
- Giulia ama il pomodoro.
- Tutti sono golosi di qualcosa.

1.  $\text{ama}(\text{maria}, \text{cibitaliani})$ .
2.  $\forall X \text{ ama}(X, \text{cibitaliani}) \Rightarrow \text{goloso}(X, \text{pasta}) \text{ or } \text{goloso}(X, \text{pizza})$
3.  $\forall X \text{ goloso}(X, \text{pasta}) \Rightarrow \text{not ama}(X, \text{zuppe})$ .
4.  $\forall X \text{ goloso}(X, \text{pizza}) \Rightarrow \text{ama}(X, \text{pomodoro})$
5.  $\forall X \text{ ama}(\text{giulia}, X) \Rightarrow \text{not ama}(\text{maria}, X)$
6.  $\text{ama}(\text{giulia}, \text{pomodoro})$
7.  $\forall X \exists Y \text{ goloso}(X, Y)$
8. Query:  $\exists X \text{ goloso}(X, \text{pasta}) \text{ and } \text{not goloso}(X, \text{pizza})$

CLAUSOLE:

1.  $\text{ama}(\text{maria}, \text{cibitaliani})$ .
2.  $\text{not ama}(X, \text{cibitaliani}) \text{ or } \text{goloso}(X, \text{pasta}) \text{ or } \text{goloso}(X, \text{pizza})$
3.  $\text{not goloso}(X, \text{pasta}) \text{ or } \text{not ama}(X, \text{zuppe})$ .
4.  $\text{not goloso}(X, \text{pizza}) \text{ or } \text{ama}(X, \text{pomodoro})$
5.  $\text{not ama}(\text{maria}, Y) \text{ or } \text{not ama}(\text{giulia}, Y)$
6.  $\text{ama}(\text{giulia}, \text{pomodoro})$
7.  $\text{goloso}(X, f(X))$ . Funzione di Skolem
8. QueryNegata:  $\text{not goloso}(X, \text{pasta}) \text{ or } \text{goloso}(X, \text{pizza})$

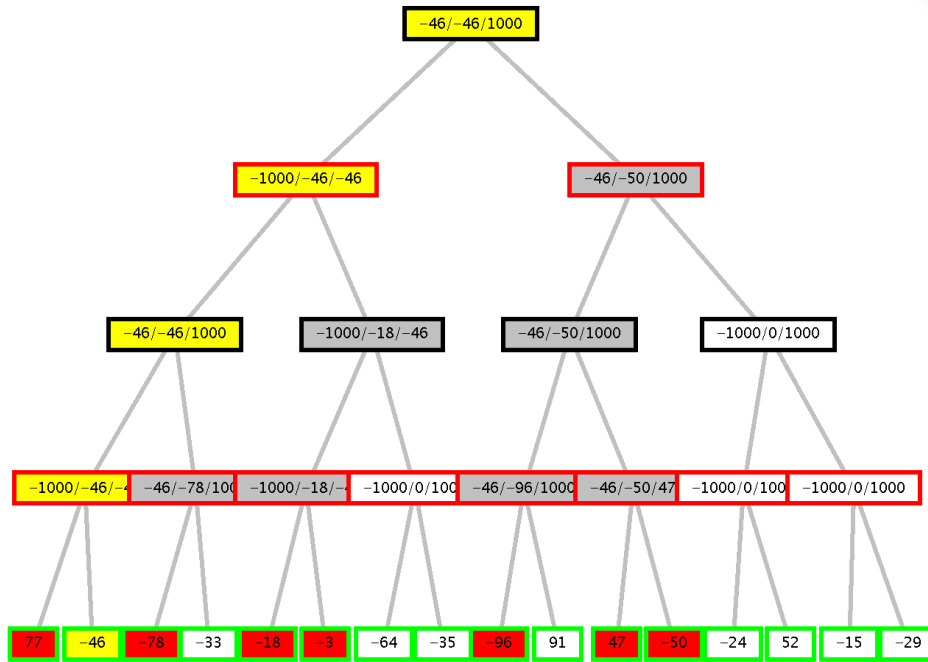
RISOLUZIONE

- 9: 5+6:  $\text{not ama}(\text{maria}, \text{pomodoro})$ .
- 10: 9+4:  $\text{not goloso}(\text{maria}, \text{pizza})$ .
- 11: 10+8:  $\text{not goloso}(\text{maria}, \text{pasta})$ .
- 12: 11+2:  $\text{not ama}(\text{maria}, \text{cibitaliani}) \text{ or } \text{goloso}(\text{maria}, \text{pizza})$
- 13: 12+10:  $\text{not ama}(\text{maria}, \text{cibitaliani})$
- 14: 13+1: contraddizione!!

### Esercizio 2

Min-max: strada in giallo, valore nodo radice -46 .

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.



### Esercizio 3

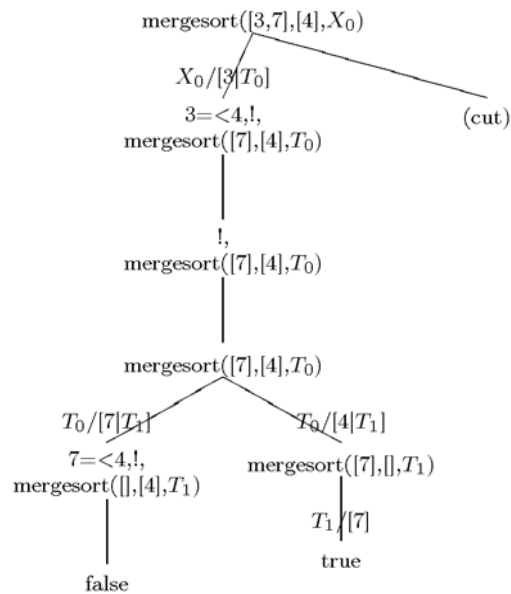
```

delete( _, [], []).
delete( X, [Y|T], [Y|L]):- X \= Y, !, delete(X,T,L).
delete( X, [_|T], L) :- delete(X,T,L).

delList( _, [], []):-!.
delList([], X, X):-!.
delList([Y|T], L1, L3) :- delete(Y,L1,L2), delList(T,L2,L3).

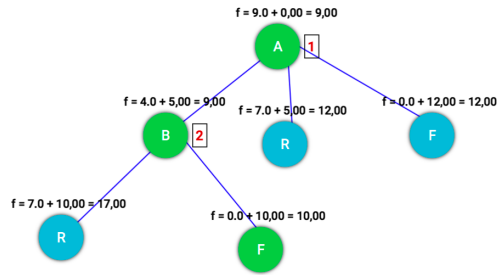
```

### Esercizio 4



### Esercizio 5

Con A\* i nodi espansi sono ABF (nodi sul cammino soluzione) e la soluzione ha costo 10 (ottimale):



Operations

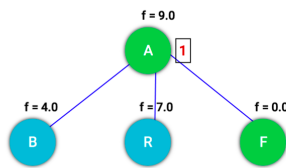
Show operations

```

1) A [f = 9.0 + 0.00 = 9.00]
2) B [f = 4.0 + 5.00 = 9.00]
/) R [f = 7.0 + 10.00 = 17.00]
/) F [f = 0.0 + 10.00 = 10.00]
/) R [f = 7.0 + 5.00 = 12.00]
/) F [f = 0.0 + 12.00 = 12.00]
    
```

Path cost: 10.0  
 Nodes expanded: 2  
 Queue size: 3  
 Max queue size: 4

Con Greedy Best First i nodi espansi sono AF (nodi sul cammino soluzione) e la soluzione ha costo 12 (non ottimale):



Operations

Show operations

```

1) A [f = 9.0]
/) B [f = 4.0]
/) R [f = 7.0]
/) F [f = 0.0]
    
```

Path cost: 12.0  
 Nodes expanded: 1  
 Queue size: 2  
 Max queue size: 3

### Esercizio 6

Vedi slide del corso per spiegare FC, PLA e FLA

Domini: A, B, C :: {1, 2, 3, 4, 5, 6, 7}

Vincoli: A > B    B > C    A > C    A = 3

FC	X1	X2	X3
	[1,2,3,4,5,6,7]	[1,2,3,4,5,6,7]	[1,2,3,4,5,6,7]
A=3, A>B, A>C	3	[1,2]	[1,2]
<b>PLA</b>			
A=3, A>B, A>C	3	[1,2]	[1,2]
B>C		[2]	[1,2]
<b>FLA</b>			
A=3, A>B, A>C	3	[1,2]	[1,2]
B>C		[2]	[1,2]
C<B		[2]	[1]