

NOTA: Consegnare la soluzione tramite un singolo file (word, pdf oppure txt)

Esercizio 1 (6 punti)

Si formalizzino le seguenti frasi in logica dei predicati:

1. Esiste almeno un appartamento che non è ristrutturato.
2. Un appartamento che è ristrutturato piace a tutti.
3. Mario vorrebbe acquistare qualunque cosa che gli piace.
4. L'appartamento rosso è ristrutturato.

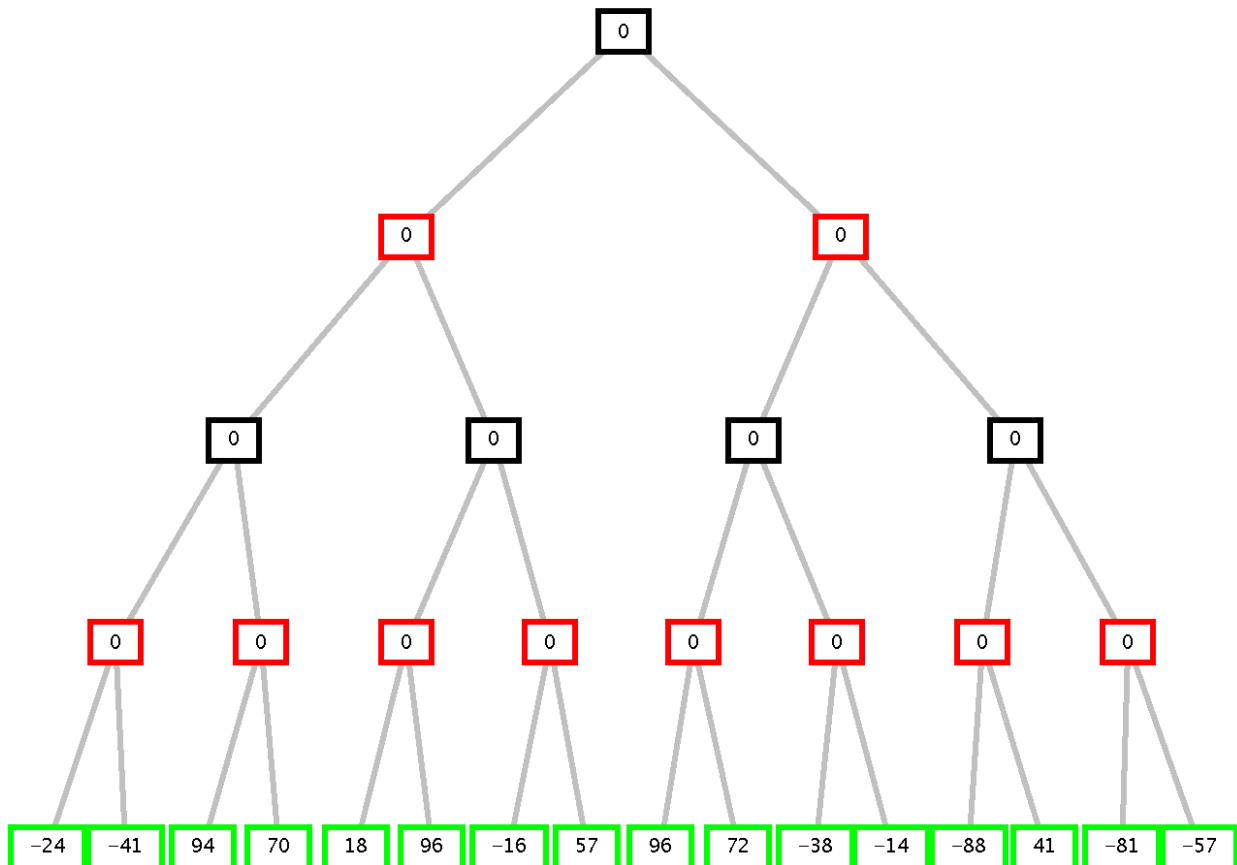
Le si trasformi in clausole usando i seguenti predicati: **appartamento(X)** (X è un appartamento), **ristrutturato(X)** (X è ristrutturato), **piace(X,Y)** (X piace a Y), **acquistare(X,Y)** (X vorrebbe acquistare Y). Si usi poi il principio di risoluzione per dimostrare che c'è un appartamento che Mario vorrebbe acquistare.

NOTA: Si riportano i simboli degli operatori e quantificatori in logica: $\forall \exists \wedge \vee \neg \rightarrow$

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 a1, o a2).
- b) 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 -24) sarà denominato a15, mentre l'ultimo arco che connette il nodo foglia più a destra (valore -57) a30.



Esercizio 3 (5 punti)

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

A::[5, 6, 7]
 B::[2, 3, 4, 5]
 C::[2, 3, 4, 5]
 D::[2, 3, 4, 5]
 A<B+3
 C<=B+1
 C>=D+2

Si applichi, durante la ricerca fino alla prima soluzione, il Forward Checking dopo ogni passo di labeling considerando, nella scelta della prossima variabile da istanziare, l'euristica **Minimum Remaining Value** (poi, a parità di cardinalità di dominio, scegliere in base all'ordine alfabetico dei nomi delle variabili). Nel labeling, per il valore da assegnare alla variabile, si considerino i valori di dominio in ordine crescente, partendo dal più piccolo.

Esercizio 4 (6 punti)

Si realizzi un predicato Prolog `lista_frequenze(L1, L2, L3)` che date due liste **L1** e **L2** deve restituire una nuova lista **L3** contenente le frequenze con cui ognuno dei valori contenuti in **L1** compare nella lista **L2**. Per "frequenza" si intende il numero di volte che un elemento **el** di **L1** compare in **L2**. Si presti attenzione al fatto che se un elemento **el** non compare mai in **L2**, la sua frequenza è **0**. Si tenga conto anche del possibile caso in cui **L2** sia una lista vuota: in tal caso la frequenza di un qualunque elemento è sempre **0**.

Al fine di implementare il predicato Prolog di cui sopra, il candidato realizzi anche un predicato `frequenza(EI, L, X)`; che calcola la frequenza **X** con cui l'elemento **EI** compare in **L**.

ESEMPLI:

?- lista_frequenze([1, 4, 2],[1, 3, 4, 1, 3, 1],L).

L = [3, 1, 0]

?- lista_frequenze([], [1, 3, 4, 1, 3, 1], L).

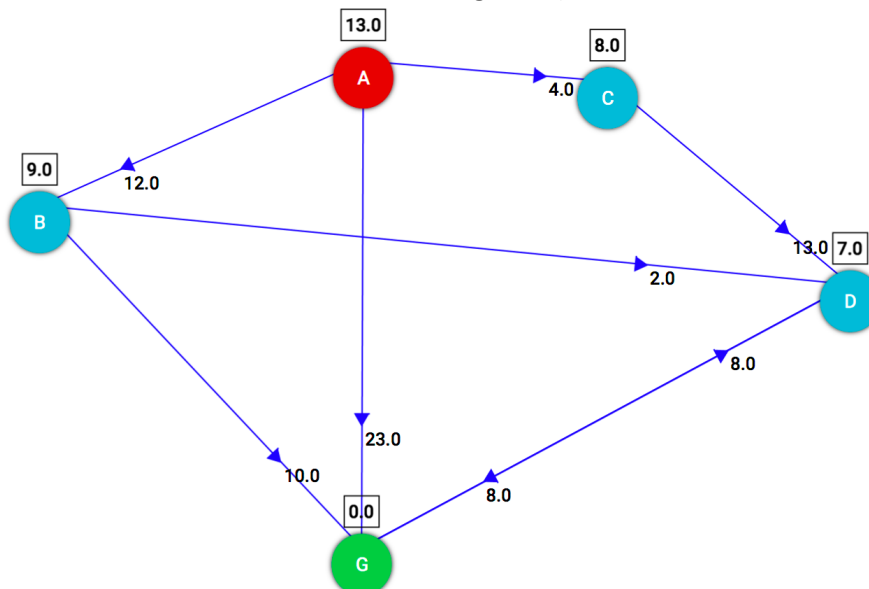
L = []

?- lista_frequenze([1,4,2],[],L).

L = [0, 0, 0]

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 (**NOTA:** l'arco D-G è bidirezionato):



Si applichi la ricerca **A*** su alberi (non tenendo quindi traccia dei nodi già visitati) e 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.

Esercizio 6 (4 punti)

Si descriva brevemente in cosa consiste la derivazione SLDNF nella programmazione logica.

Esercizio 1

1. $\exists X (appartamento(X) \wedge \neg ristrutturato(X))$
2. $\forall X \forall Y (appartamento(X) \wedge ristrutturato(X) \rightarrow piace(X,Y))$
3. $\forall X (piace(X,mario) \rightarrow acquistare(mario,X))$
4. $appartamento(rosso) \wedge ristrutturato(rosso)$

Goal: $\exists X appartamento(X) \wedge acquistare(mario, X)$

Clausole:

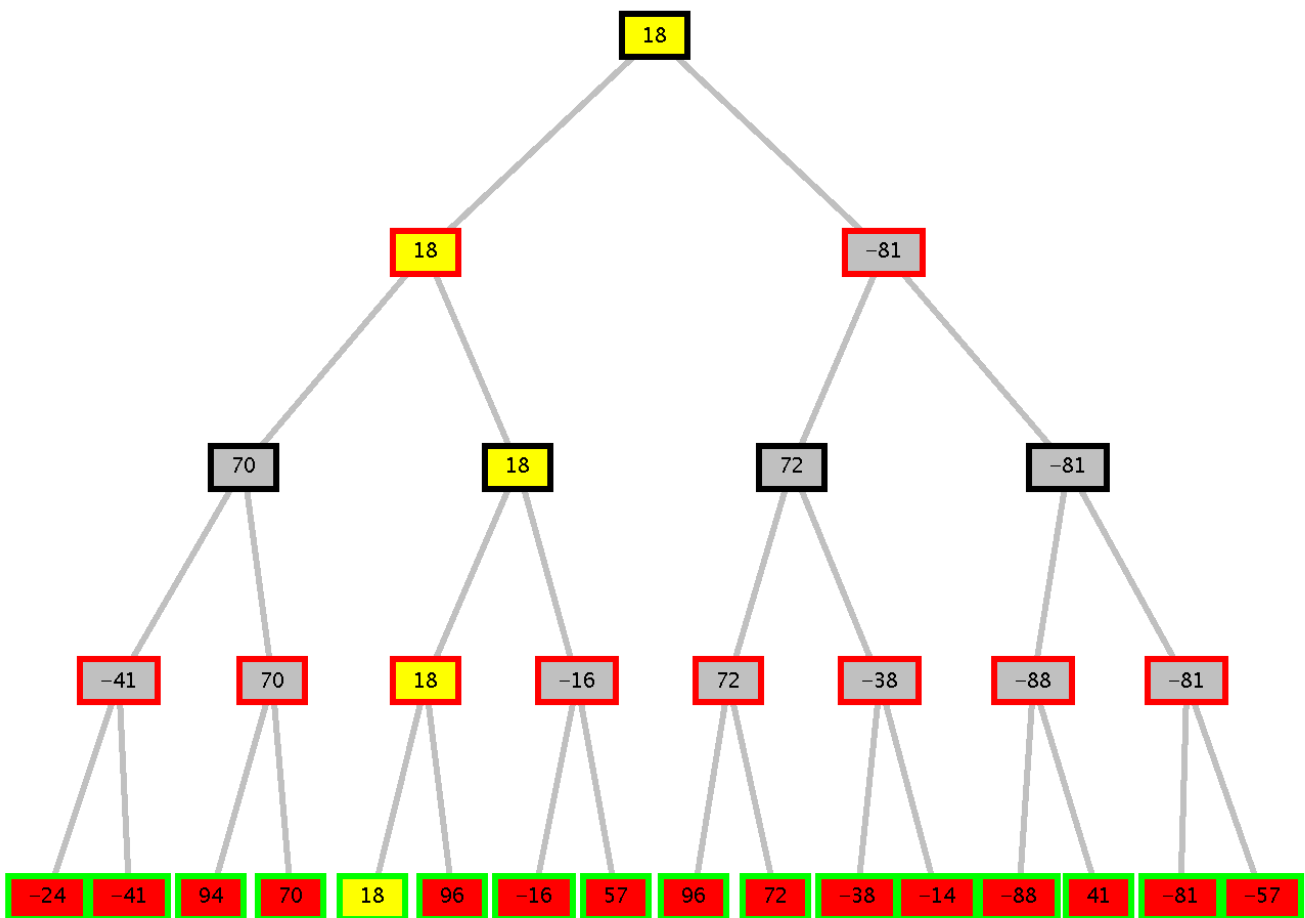
- 1a. $appartamento(c1)$
- 1b. $\neg ristrutturato(c1)$
2. $\neg appartamento(X) \vee \neg ristrutturato(X) \vee piace(X,Y)$
3. $\neg piace(X,mario) \vee acquistare(mario,X)$
- 4a. $appartamento(rosso)$
- 4b. $ristrutturato(rosso)$
- GNeg: $\neg appartamento(X) \vee \neg acquistare(mario,X)$

Risoluzione:

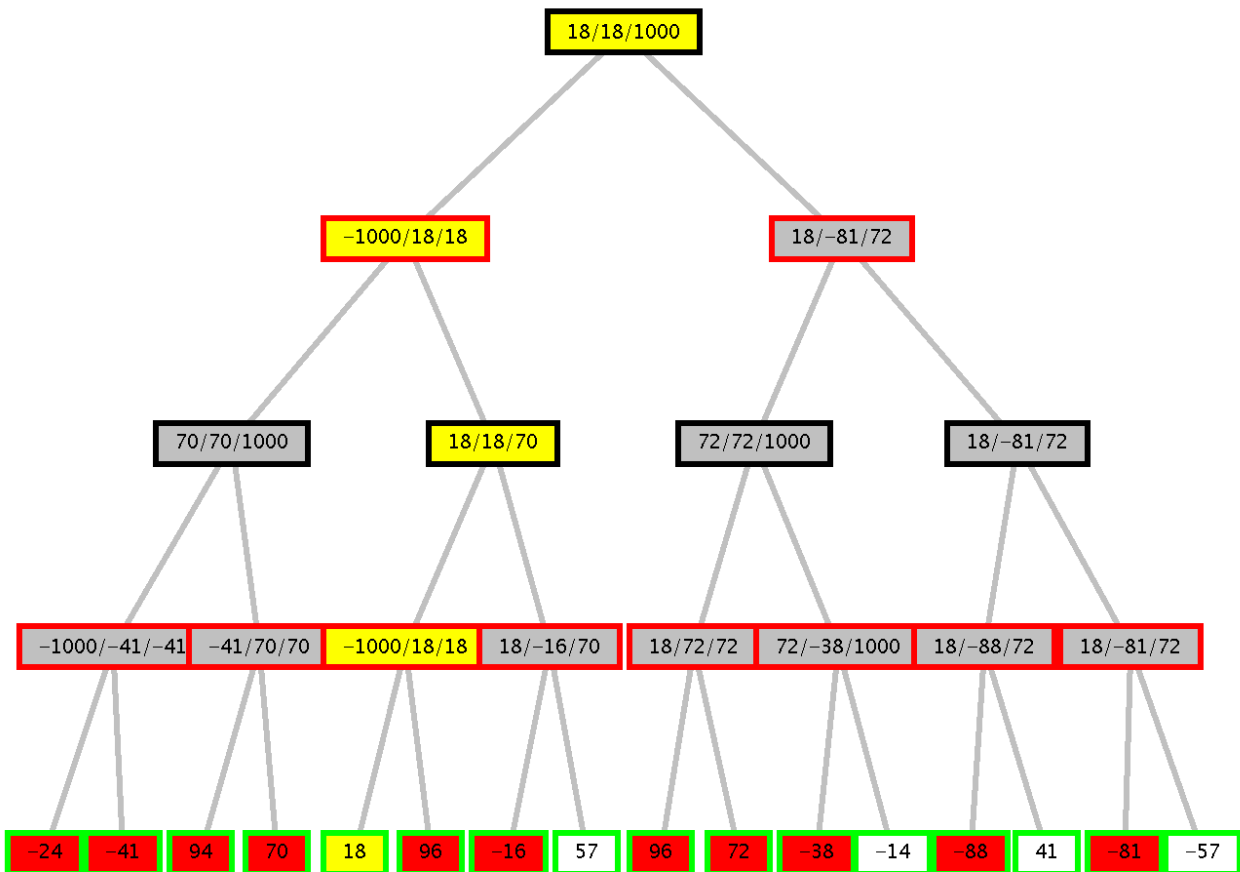
- 5.: GNeg+4a: $\neg acquistare(mario,rosso)$
- 6.: 5 + 3: $\neg piace(rosso,mario)$
- 7.: 6 + 2: $\neg appartamento(rosso) \vee \neg ristrutturato(rosso)$
- 8.: 7 + 4a: $\neg ristrutturato(rosso)$
- 9.: 8 + 4b: Contraddizione! Clausola vuota

Esercizio 2

Min-max: strada in giallo – valore nodo radice 18, ramo a1.



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, a28, a30.
 Scelta per il ramo a1, valore propagato 18.

Esercizio 3

Con euristica MRV:

	A	B	C	D
	[5..7]	[2..5]	[2..5]	[2..5]
Labeling	A=5	[2...5]	[2...5]	[2...5]
FC	A=5	[3...5]	[2...5]	[2...5]
Labeling	A=5	B=3	[2...5]	[2...5]
FC	A=5	B=3	[2...4]	[2...5]
Labeling	A=5	B=3	C=2	[2...5]
FC	A=5	B=3	C=2	Fail
Backtracking e Labeling	A=5	B=3	C=3	[2...5]
FC	A=5	B=3	C=3	Fail
Backtracking e Labeling	A=5	B=3	C=4	[2...5]
FC	A=5	B=3	C=4	[2]
Labeling	A=5	B=3	C=4	D=2
Soluzione:	A=5	B=3	C=4	D=2

Esercizio 4

```

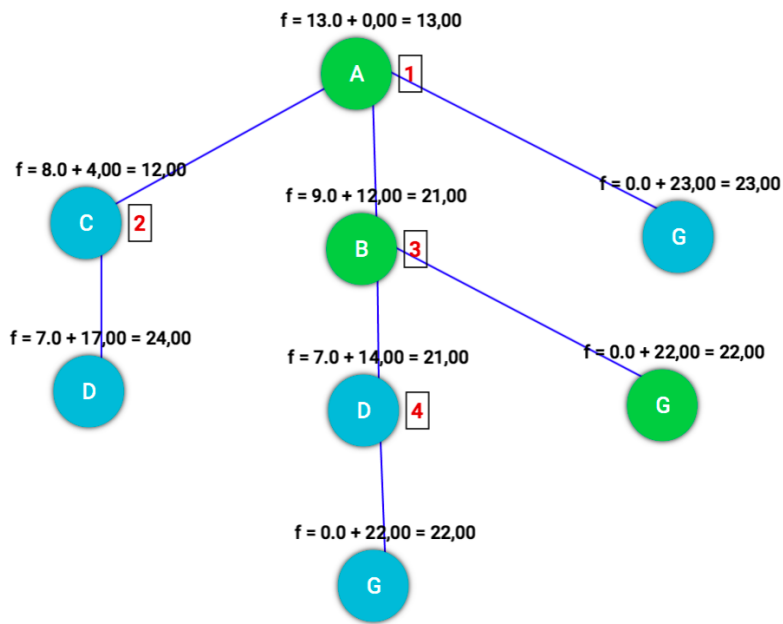
lista_frequenze([],_,[]):-!.
lista_frequenze([X|Y],Z,[X1|Y1]):- frequenza(X,Z,X1), lista_frequenze(Y,Z,Y1).

frequenza(_,[],0).
frequenza(X,[X|Y],Z):-!,frequenza(X,Y,Z1), Z is Z1+1.
frequenza(X,[_|Y],Z):-frequenza(X,Y,Z).

```

Esercizio 5

Con A*, i nodi espansi sono ABG, la soluzione trovata ABG è ottimale perché l'euristica è ammissibile e il costo della soluzione trovata è 22:



▼ Operations

Show operations

- 1) A [f = 13.0 + 0,00 = 13,00]
- 2) C [f = 8.0 + 4,00 = 12,00]
- 3) B [f = 9.0 + 12,00 = 21,00]
- 4) D [f = 7.0 + 14,00 = 21,00]
- /) D [f = 7.0 + 17,00 = 24,00]
- /) G [f = 0.0 + 22,00 = 22,00]
- /) G [f = 0.0 + 22,00 = 22,00]
- /) G [f = 0.0 + 23,00 = 23,00]

Path cost: 22.0
 Nodes expanded: 4
 Queue size: 3
 Max queue size: 4

Esercizio 6

Si vedano le slide del corso.