

**COMPITO DI FONDAMENTI DI INTELLIGENZA ARTIFICIALE**  
**INTELLIGENZA ARTIFICIALE (v.o.) – PARTE I**

**10 Gennaio 2008 (Tempo a disposizione 2h e 1/2; su 32 punti)**

**Esercizio 1 (punti 7)**

Si rappresenti in logica dei predicati del I ordine la frase:

“Soltanto se (se e solo se) un’automobile ha benzina può funzionare”.

Si mostri mediante il principio di risoluzione, quali delle seguenti query sono derivabili.

- 1) Se un’automobile ha la benzina allora funziona
- 2) Se un’automobile non funziona, allora non ha la benzina
- 3) Se un’automobile funziona, allora ha la benzina

**Esercizio 2 (punti 6)**

Si consideri il seguente programma Prolog:

```
elems([], []).  
elems([H|T], [H|L]) :- not(H=[]), not(H=[A|B]), !,  
    elems(T,L).  
elems([H|T], L) :- elems(T,L).
```

Si disegni l’albero SLDNF relativo all’invocazione `elems([2, [3,4], X], E)` e si dica qual è la risposta calcolata.

**Esercizio 3 (punti 4)**

Si scriva un programma Prolog che data una lista `List1` composta da liste di interi, restituisca in uscita una lista `List2` di interi che all’*i*-esimo posto contiene la somma di tutti gli elementi della *i*-esima lista di `List1`.

Esempi:

```
?- listsum([[4,5], [3,7], [1,9]], L) yes L= [9,10,10].  
?- listsum([[4,5,3,7], [2,4]], L) yes L= [21,6].
```

**Esercizio 4 (punti 7)**

Il Signor Gedeone è davvero un patito di enigmistica. A un amico che gli chiede quanti giorni di vacanza abbia trascorso al mare la scorsa estate, risponde in questa maniera: “Mia moglie Miranda e mio figlio Filippo sono stati al mare con me, ma Miranda è stata la prima dei tre ad arrivare al mare ed è restata 8 giorni più di Filippo; Filippo d’altra parte è arrivato al mare 3 giorni prima di me e se ne è andato 5 giorni prima di me. Io sono arrivato per ultimo e sono restato fino a quando è restata Miranda, che alla fine si è fatta ben tre settimane di mare!”.

Si modelli il CSP suggerito dal Signor Gedeone, sapendo che nessuno dei tre componenti ha passato al mare più di tre settimane e ciascuno è stato al mare almeno 10 giorni.

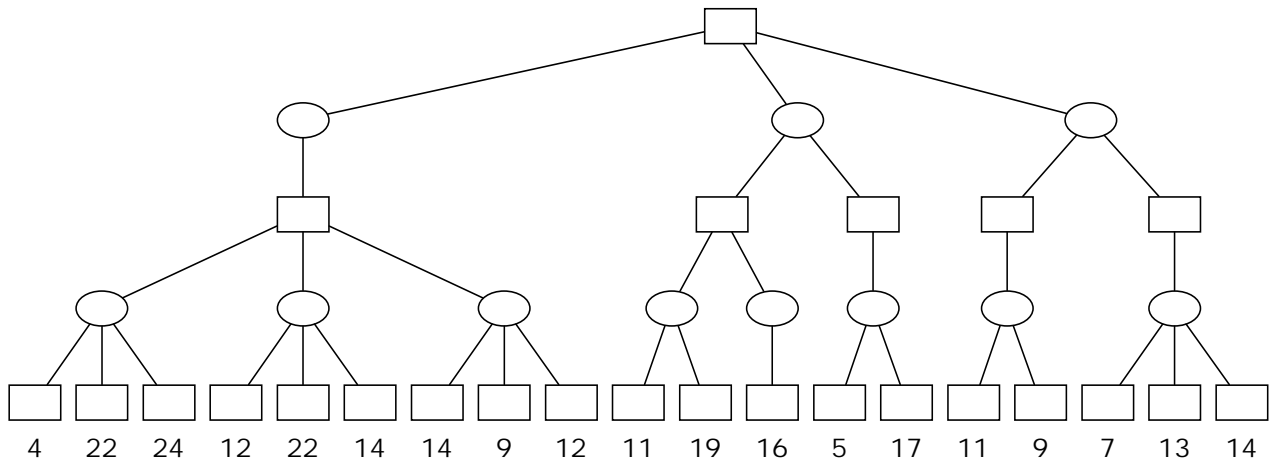
Si semplifichino i vincoli *n*-ari ricavandone vincoli binari, ove possibile.

Si applichi l’arc-consistenza al CSP modellato.

Si trovi poi la prima soluzione applicando il forward checking. Si considerino le variabili soggette a labeling applicando il First Fail Principle e i valori dei domini dal maggiore al minore, con domini iniziali [0..21] per ciascuna variabile (l’ipotesi: “Miranda è stata la prima ad arrivare al mare” corrisponde al valore di dominio 0).

### Esercizio 5 (punti 5)

Si consideri il seguente albero di gioco:



in cui i punteggi sono dal punto di vista del primo giocatore (Max). Si mostri il ramo scelto dall'algoritmo min-max. Si mostrino poi i tagli alfa-beta.

### Esercizio 6 (punti 3)

Si spieghi come viene trattata la negazione in Prolog, sottolineandone problemi e limitazioni. Si mostri la sua realizzazione in termini dei predicati cut e fail.

## SOLUZIONE

### Esercizio 1

A0)  $\forall A (ha(A, benzina) \Leftrightarrow funziona(A))$

Q1)  $\forall A (ha(A, benzina) \Rightarrow funziona(A))$

Q2)  $\forall A (\neg funziona(A) \Rightarrow \neg ha(A, benzina))$

Q3)  $\forall A funziona(A) \rightarrow ha(A, benzina)$

#### Q1:

$\neg Q1: \neg(\forall A (ha(A, benzina) \Rightarrow funziona(A)))$

Trasformazione in clausole di Horn:

A0:

- $\neg ha(A, benzina) \vee funziona(A)$
- $ha(A, benzina) \vee \neg funziona(A)$

$\neg Q1:$

- $ha(c1, benzina)$
- $\neg funziona(c1)$

Risoluzione

1.  $\neg ha(A, benzina) \vee funziona(A)$
2.  $\neg funziona(c1)$
3.  $ha(c1, benzina)$
4. (da , **Error! Reference source not found.**)  $funziona(c1)$
5. (da , )  $\square$

#### Q2:

$\neg Q2: \neg(\forall A (\neg funziona(A) \Rightarrow \neg ha(A, benzina)))$

Trasformazione in clausole di Horn:

- $\neg funziona(c1)$
- $ha(c1, benzina)$

Risoluzione

1.  $\neg ha(A, benzina) \vee funziona(A)$
2.  $\neg funziona(c1)$
3.  $ha(c1, benzina)$
4. (da , **Error! Reference source not found.**)  $funziona(c1)$
5. (da , )  $\square$

#### Q3:

$\neg Q3: \neg(\forall A (funziona(A) \Rightarrow ha(A, benzina)))$

Trasformazione in clausole di Horn:

- $funziona(c1)$
- $\neg ha(c1, benzina)$

## Risoluzione

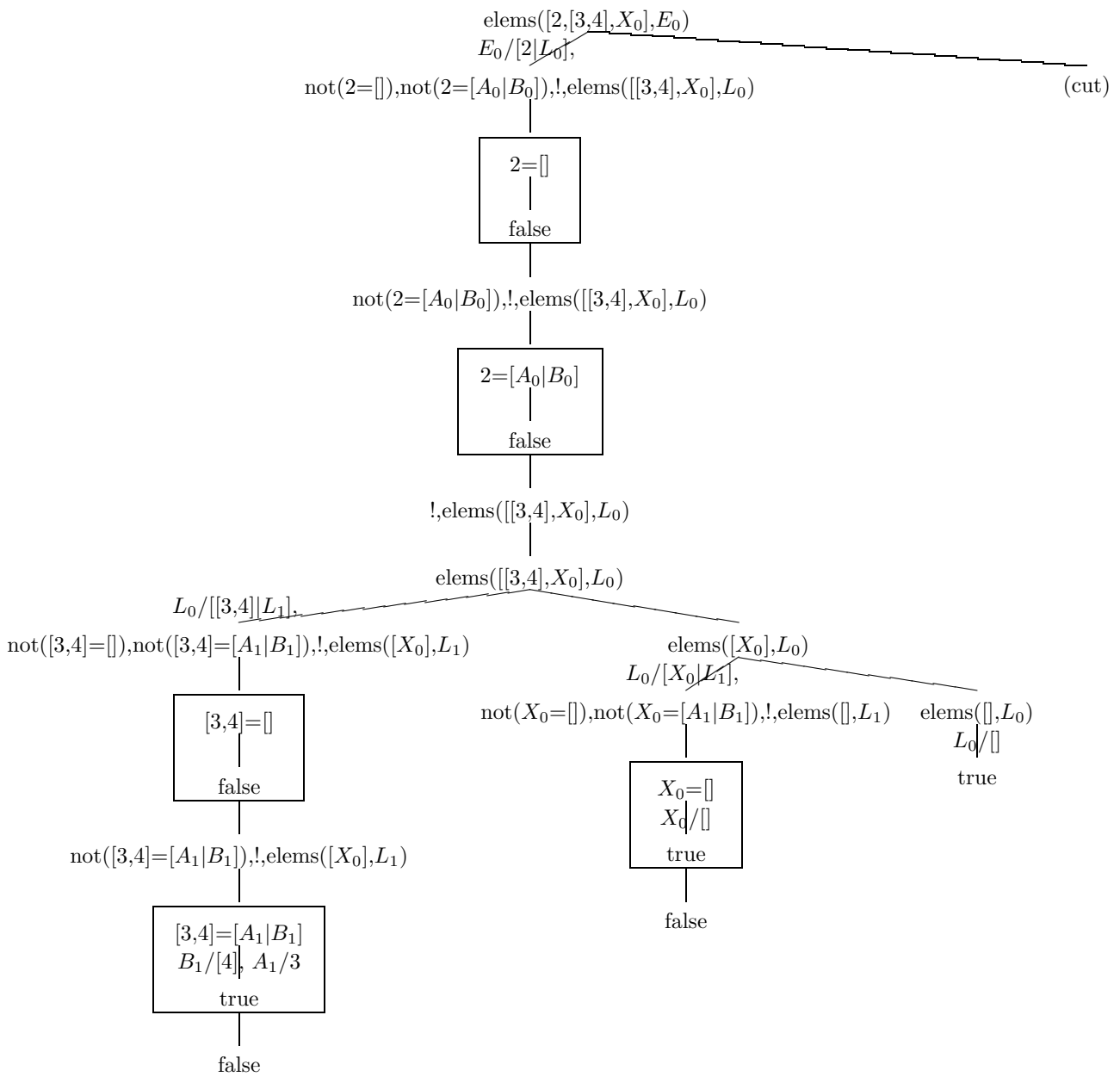
1.  $ha(A, benzina) \vee \neg funziona(A)$
2.  $\neg ha(c1, benzina)$
3.  $funziona(c1)$
4. (da , **Error! Reference source not found.**)  $ha(c1, benzina)$
5. (da , )  $\square$

La 1) è vera nella teoria data dall'assioma 0) perché è una parte della doppia implicazione.

La 2) è vera perché vale la contrappositiva

La 3) è vera perché è una parte della doppia implicazione.

## Esercizio 2



La risposta calcolata è  $E/[2].$

**Esercizio 3:**

```
listsum([], []).  
listsum([A], [S]) :- ! sum(A,S).  
listsum([A|B], [S|L]) :- sum(A,S), listsum(B, L).  
  
sum([], 0) :- !.  
sum([X], X) :- !.  
sum([X|Y], N1) :- sum(Y, N2), N1 is X+N2.
```

**Esercizio 4**

Variabili intere (pedice 1, inizio vacanza, 2 fine):

```
F1=G1-3  
F2=G2-5  
G1>F1>M1  
G2>=G1+10  
M2-M1=21  
M1=0  
F2>=F1+10  
M2-M1=21=F2-F1+8  
M2=G2
```

Semplificando:

```
F1=G1-3  
F2=G2-5  
G1>F1>0  
G2>=G1+10  
F2>=F1+10 (inutile)  
F2=F1+13  
M2-M1=21  
M2=G2
```

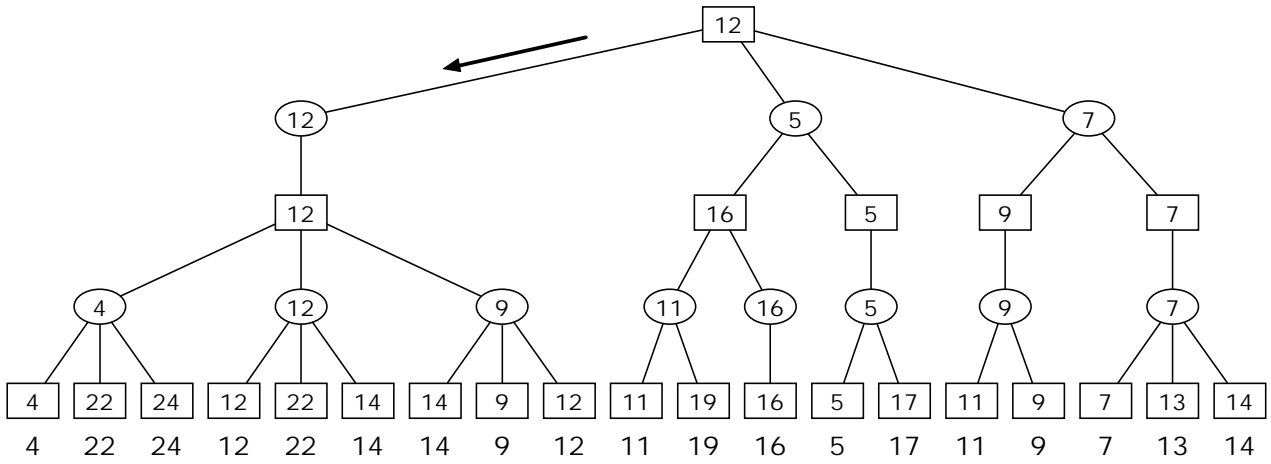
Arc consistenza:

```
M1::[0]  
M2::[21]  
F1::[3]  
F2::[16]  
G1::[6]  
G2::[21]
```

La fase successiva non fa altro che assegnare alle variabili l'unico valore nel dominio.

### Esercizio 5

Min-max:



Tagli alfa-beta:

