FONDAMENTI DI INTELLIGENZA ARTIFICIALE

13 Gennaio 2010 – Tempo a disposizione 2h – Risultato 32/32 punti

Esercizio 1 (punti 6)

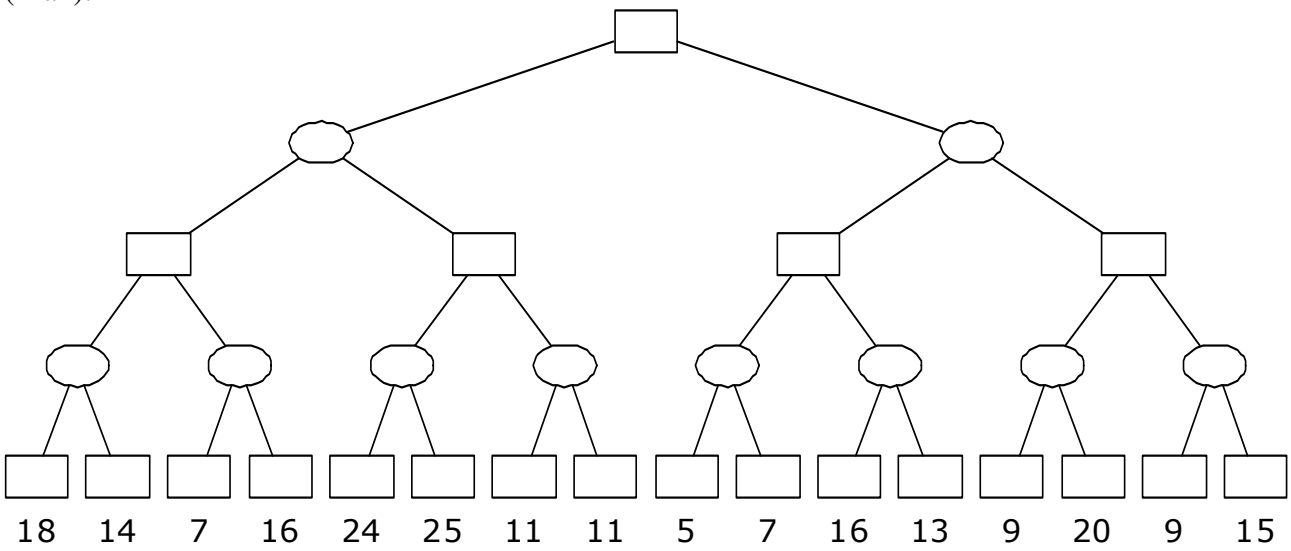
Si modellino in logica dei predicati del I ordine le seguenti frasi (utilizzando i predicati motociclista/1, guida/2, duro/1, yuppie/1, avvocato/1):

1. *Ogni motociclista guida qualcosa che è una Harley o una BMW.*
2. *Chiunque guidi una Harley è un duro.*
3. *Chiunque guidi una BMW è uno yuppie.*
4. *Ogni yuppie è un avvocato.*
5. *John è un motociclista.*
6. *John non è un avvocato.*

a) Le si trasformi in logica a clausole e si dimostri, applicando il principio di risoluzione, che:
Q: John è un duro.

Esercizio 2 (punti 5)

Si consideri il seguente albero di gioco, dove i punteggi sono dal punto di vista del primo giocatore (Max):



Si mostri come gli algoritmi min-max e alfa-beta risolvono il problema.

Esercizio 3 (punti 7)

Si consideri il seguente programma Prolog:

```
minimo([H|T],Min,Rest):-
    minimo(T,H,Min,Rest).
minimo([],Min,Min,[]).
minimo([H|T],MinIn,MinOut,[H|Rest1]):-
        MinIn<H,!,
        minimo(T,MinIn,MinOut,Rest1).
minimo([H|T],MinIn,MinOut, [MinIn|Rest1]):-
        minimo(T,H,MinOut,Rest1).
```

Si disegni l'albero SLD relativo al goal

```
?-minimo([10,20,1,3,5,0],X,R).
```

Esercizio 4 (punti 7)

Si trovi un ordinamento (totale) per le variabili A, B, C, D, E che soddisfi i seguenti vincoli:

D *sta tra* A *e* B
E *e* D *stanno dopo* C
E *e* A *stanno tra* C *e* D

Quando si dice che *x sta tra y e z* si intende che *x segue y* e *x precede z.*
Si risolva il problema con le tecniche di soddisfacimento di vincoli utilizzando Arc Consistency, e le euristiche MRV e del grado per la scelta delle variabili. Nel caso ancora di scelte multiple, utilizzare come ultimo criterio di scelta l'ordine alfabetico delle variabili.

Per agevolarci nella correzione si chiede di seguire in dettaglio i seguenti passi:

a. Si formuli il problema come un problema di soddisfacimento di vincoli indicando quali sono le variabili, i relativi domini ed esprimendo esplicitamente tutti i vincoli.
b. Si disegni il grafo dei vincoli (tutti i vincoli sono esprimibili come vincoli binari).
c. Si evidenzi il risultato di un passo iniziale di controllo di consistenza degli archi.
d. Si proceda a trovare la soluzione utilizzando per la scelta della variabile l'euristica MRV (o *minimum remaining value*) e, a parità di valutazione, l'euristica del grado (o *most constraining variable)* che suggerisce di scegliere la variabile che è coinvolta nel maggior numero possibile di vincoli; nel caso ancora di scelte multiple, utilizzare l'ordine alfabetico delle variabili; dopo ogni assegnamento ristabilire la consistenza degli archi ed evidenziare il risultato ottenuto ridisegnando il grafo.

Esercizio 5 (punti 5)

Si scriva un predicato Prolog `ordina(L1,Lord)` che ordina gli elementi della lista di interi `L1` e fornisce la sua versione ordinata in `Lord` realizzando l'algoritmo di *naive sort* (suggerimento: nella soluzione si può utilizzare il predicato `minimo` dell'esercizio 3).

Esercizio 6 (punti 2)

Si dia la descrizione di ricerca iterative deepening, anche in pseudo-codice. Si discutano le sue proprietà (in termini di completezza, ottimalità e complessità sia spaziale sia temporale).

SOLUZIONE:

Esercizio 1

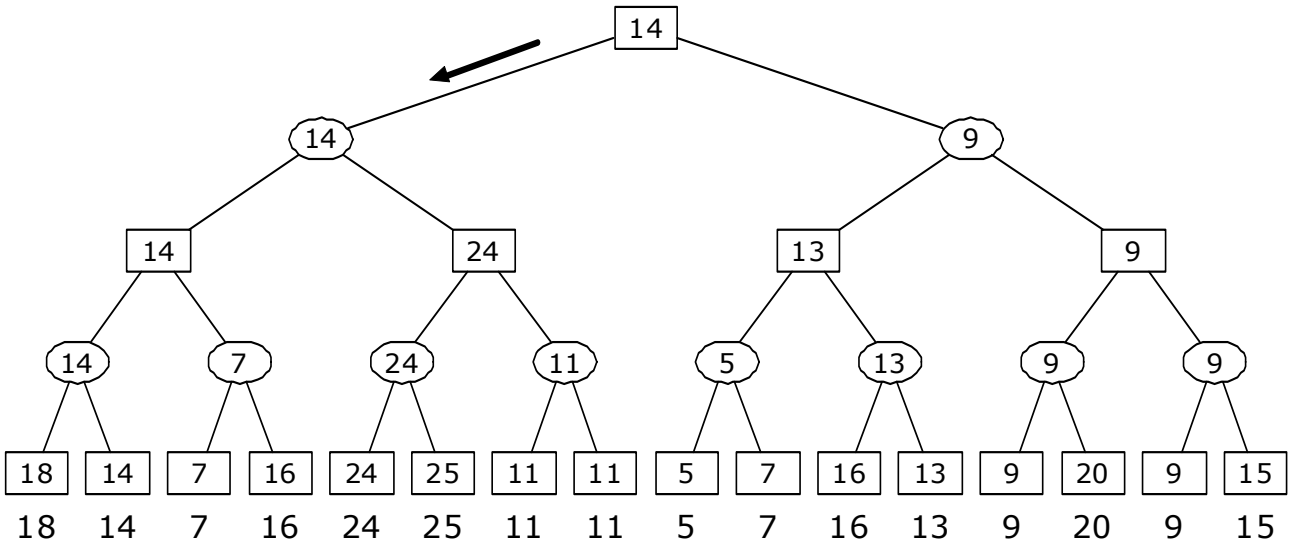
∀X motociclista(X) ---> guida(X, harley) ∨ guida(X, bmw).
∀X guida(X, harley) ---> duro(X).
∀X guida(X, bmw) ---> yuppie(X).
∀X yuppie(X) ---> avvocato(X).
motociclista(john).
not avvocato(john).
GOAL: duro(john).

C1: not motociclista(X) ∨ guida(X, harley) ∨ guida(X, bmw).
C2: not guida(X, harley) ∨ duro(X).
C3: not guida(X, bmw) ∨ yuppie(X).
C4: not yuppie(X) ∨ avvocato(X).
C5: motociclista(john).
C6: not avvocato(john).
C7 (GOAL negato): not duro(john).

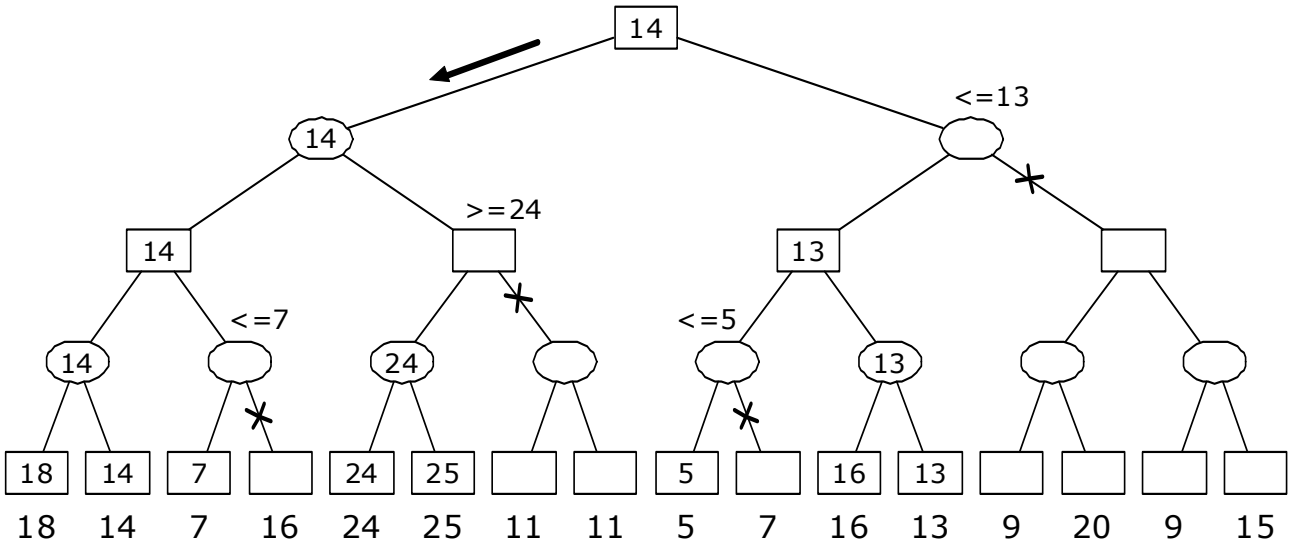
C8 (C7+C2): not guida(john, harley).
C9 (C8 + C1): not motociclista(john) ∨ guida(john, bmw).
C10 (C9 + C5): guida(john, bmw).
C11 (C10 + C3): yuppie(john).
C12 (C11 + C4): avvocato(john).
C13 (C12 + C6): []

Esercizio 2

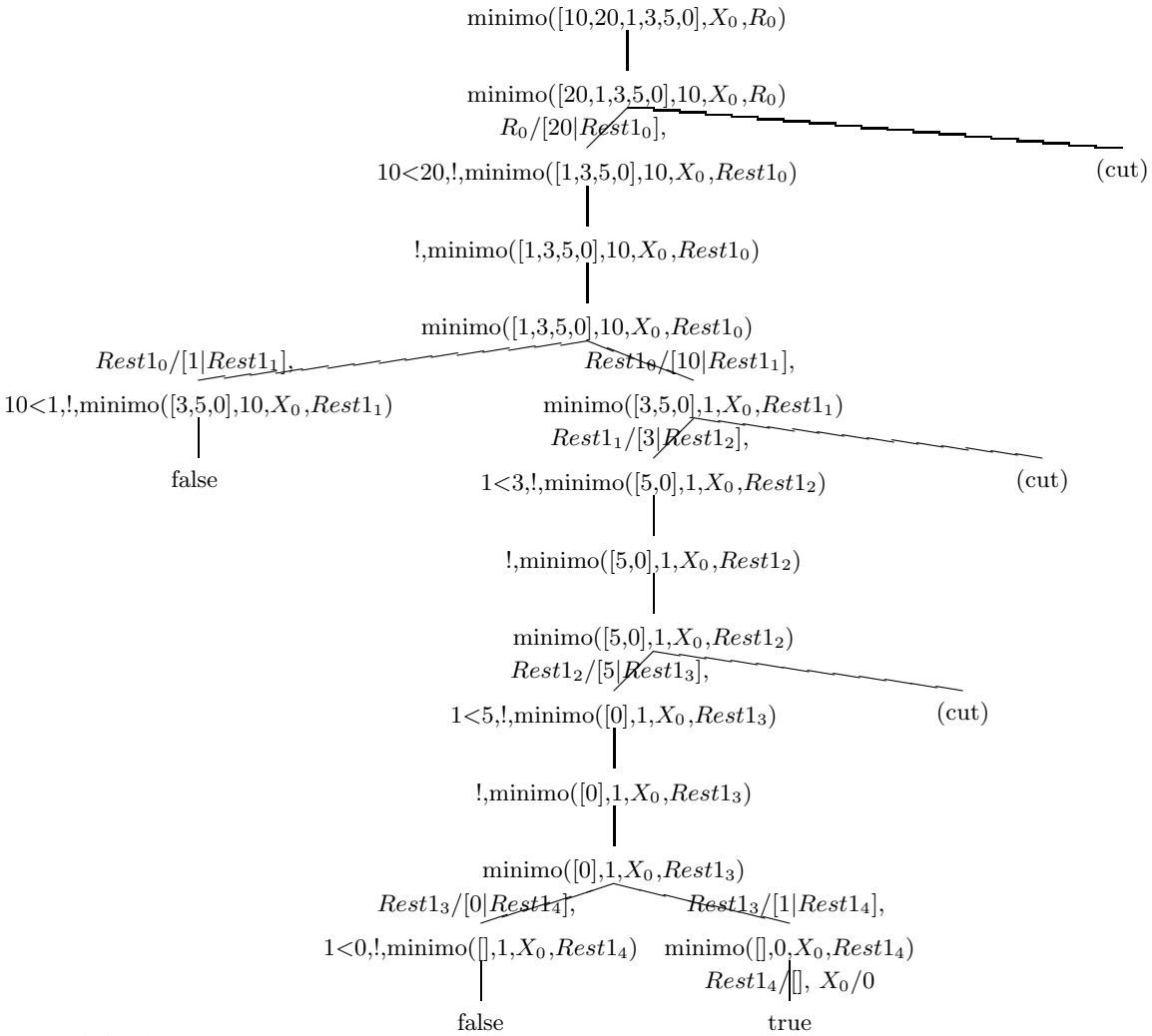
<u>min-max:</u>



Tagli alfa-beta:



Esercizio 3



Esercizio 4

a)

Le variabili sono:

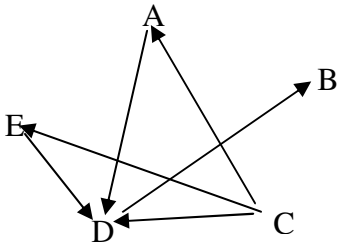
A, B, C, D, E

I relativi domini sono le posizioni possibili: {1, 2, 3, 4, 5}

I vincoli:

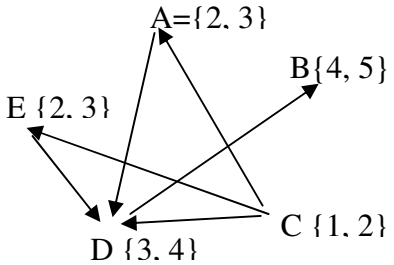
A≠B≠C≠D≠E
A < D; D < B
C < E; C < D
C < E; E < D; C < A; A < D

b) Grafo dei vincoli (non si indicano i vincoli ≠ per semplicità)

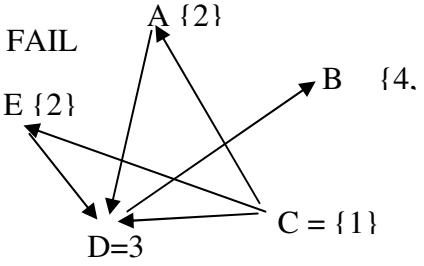


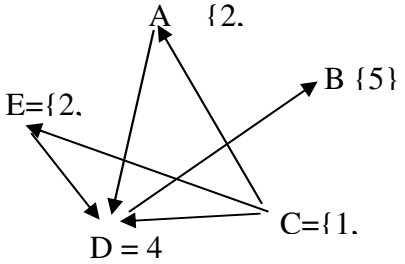
c) dopo un passo iniziale di consistenza degli archi

A ∈ {2, 3}
B ∈ {4, 5}
C ∈ {1, 2}
D ∈ {3, 4}
E ∈ {2, 3}

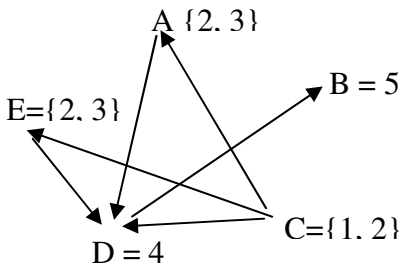


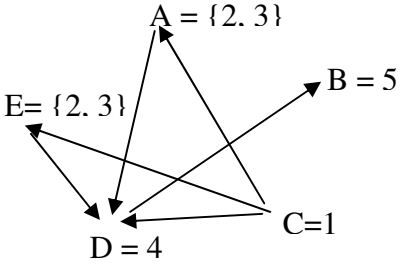
d) Scelgo la variabile con grado maggiore, D; provo a assegnare 3, ma fallisco, poi assegno 4.



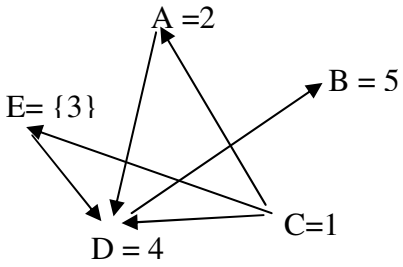


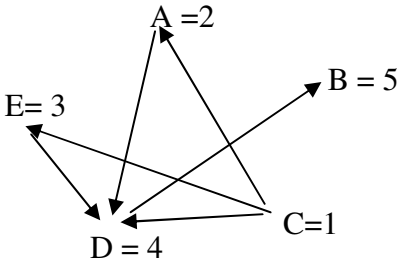
Scelgo B, che ha 1 solo valore e dopo la propagazione, tra A, E e C che hanno tutte due valori, scelgo C perché ha grado maggiore:





Tra A ed E, entrambe con due valori e stesso grado, scelgo A (ordine alfabetico) ed infine E:





Un ordinamento che soddisfa i vincoli è pertanto: C, A, E, D, B

Esercizio 5

```
ordina([],[]):- !.
ordina(L,[Hord|Tord]):-
    minimo(L,Hord,Rest),
    ordina(Rest,Tord).

minimo([H|T],Min,Rest):-
    minimo(T,H,Min,Rest).
% Calcolo del minimo: versione tail-recursive
minimo([],Min,Min,[]).
minimo([H|T],MinIn,MinOut,Rest):-
    MinIn<H,!,
    Rest=[H|Rest1],
    minimo(T,MinIn,MinOut,Rest1).
minimo([H|T],MinIn,MinOut,Rest):-
    Rest=[MinIn|Rest1],
    minimo(T,H,MinOut,Rest1).
```