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

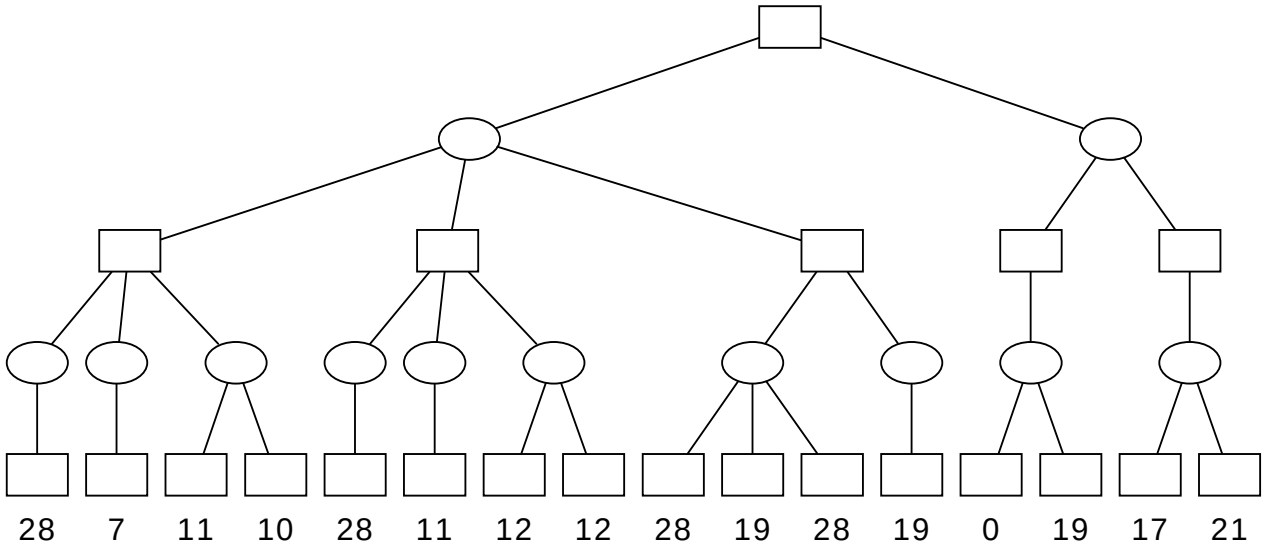
11 Settembre 2008 (Tempo a disposizione 2h ; su 32 punti)

Esercizio 1 (punti 3)
Dire quali tra le seguenti formule nella logica del primo ordine sono una rappresentazione adeguata della frase:

I clown hanno il naso rosso.

Discutere e motivare le risposte.
a1. $\forall x\ [clown(x) \rightarrow rosso(nasoDi(x))]$
a2. $\forall x\ rosso(nasoDi(clown(x)))$
a3. $\forall x\ \forall y\ [clown(x) \wedge naso(x,y) \rightarrow rosso(y)]$

Esercizio 2 (punti 6)
Si consideri il seguente albero di gioco, dove i punteggi sono dal punto di vista del primo giocatore (Max):



Si mostri come l'algoritmo min-max risolve il problema. Si mostrino poi i tagli alfa-beta.

Esercizio 3 (punti 6)
Si consideri il seguente programma Prolog, che serve a calcolare il numero di elementi totali in una lista di liste:

```
numel(L,N):-numel(L,0,N).
numel([],N,N):-!.
numel([H|T],Ni,No):-!,numel(H,Ni,Nt),numel(T,Nt,No).
numel(E,Ni,No):-No is Ni+1.
```

Si disegni l'albero SLD relativo al goal `numel([1,[]],N).`

Esercizio 4 (punti 5)
Si scriva un programma Prolog `no_dupl(Xs,Ys)` che è vero se `Ys` è la lista (senza duplicazioni) degli elementi che compaiono nella lista `Xs`. Nella lista `Ys` gli elementi compaiono nello stesso ordine di Xs ed, in caso di elementi duplicati, si manterra` l'ultima occorrenza.
Esempi:

```
?-no_dupl([a,b,a, d],[b,a,d]).
yes
?-no_dupl([a,b,a,c,d,b,e], L).
```

```
yes  L=[a,c,d,b,e]
```

Esercizio 5 (punti 9)

Si vuole colorare le celle di una griglia 3x3 con due colori (R e B) in modo da minimizzare il numero di celle adiacenti dello stesso colore, avendo a disposizione un operatore che inverte il colore di una cella (da R a B, o da B ad R).

1. Si mostrino gli stati che risolvono il problema avendo 0 celle adiacenti dello stesso colore.
2. Data una funzione di valutazione che conta il numero di coppie di caselle adiacenti dello stesso colore, si tracci l'andamento dell'algoritmo di *hill-climbing* (o meglio *hill-descending*) a partire dalla seguente configurazione iniziale:
   a. R R R
   b. B R R
   c. B B B
3. La funzione di valutazione definita precedentemente potrebbe essere utilizzata come un'euristica ammissibile per il problema applicando una strategia di ricerca A*? Motivare la risposta.

Esercizio 6 (punti 3)

Si introduca il concetto di unificazione e di sostituzione unificatrice.

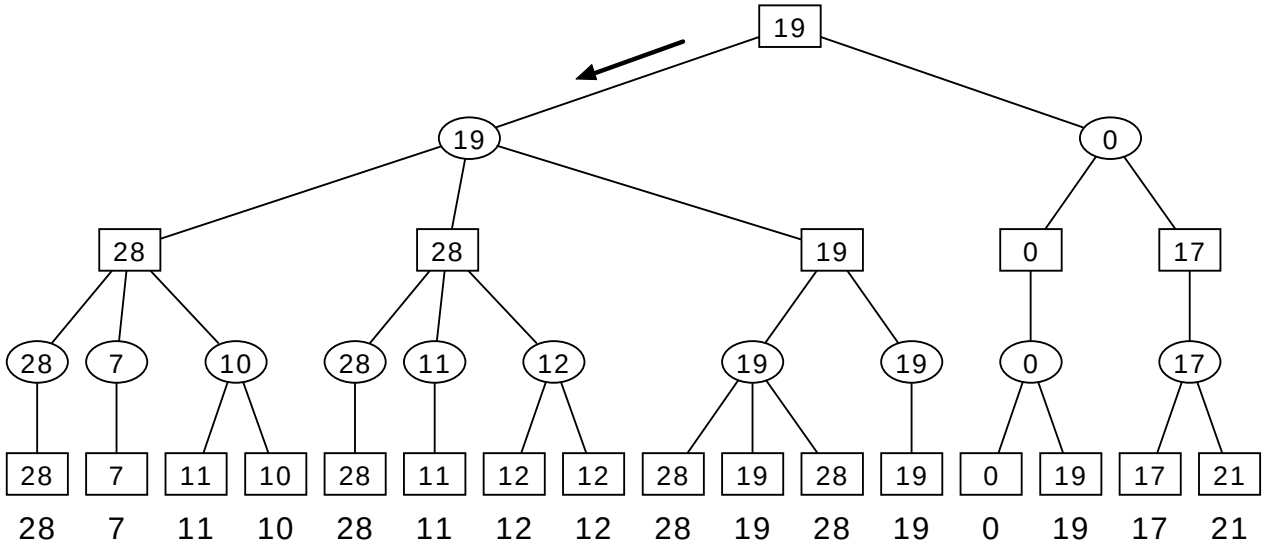
SOLUZIONE

Esercizio 1

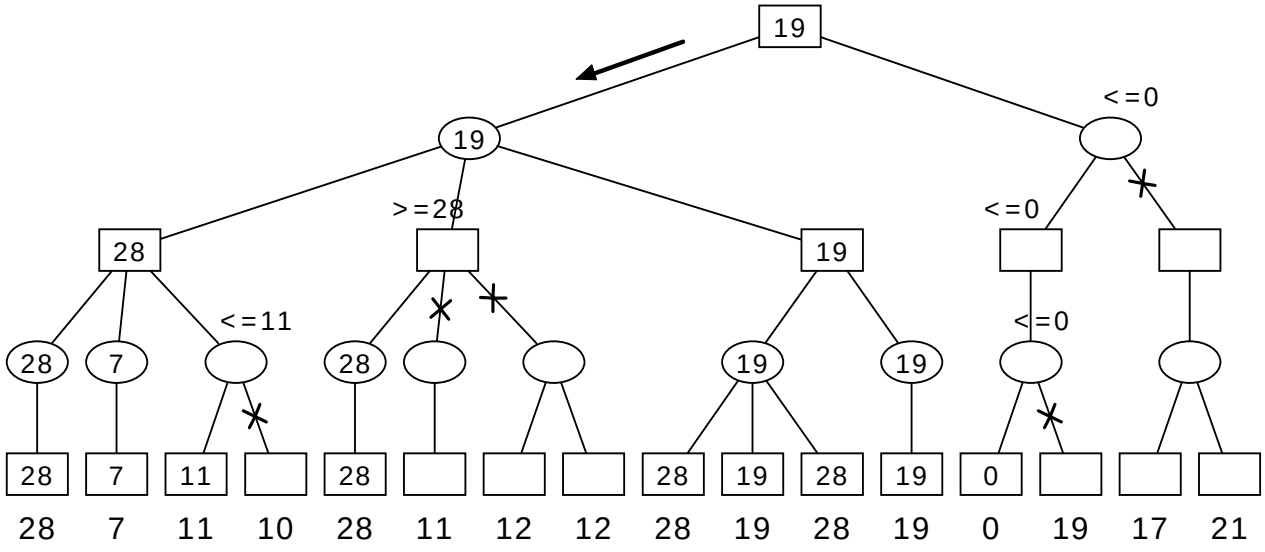
a1. corretta se *clown* e *rosso* sono due predicati unari e *nasoDi* è una funzione unaria.
a2. errata: *clown* deve essere un simbolo di predicato e non può comparire in un termine.
a3. corretta, a differenza di a1 usa tutti simboli di predicato.

Esercizio 2

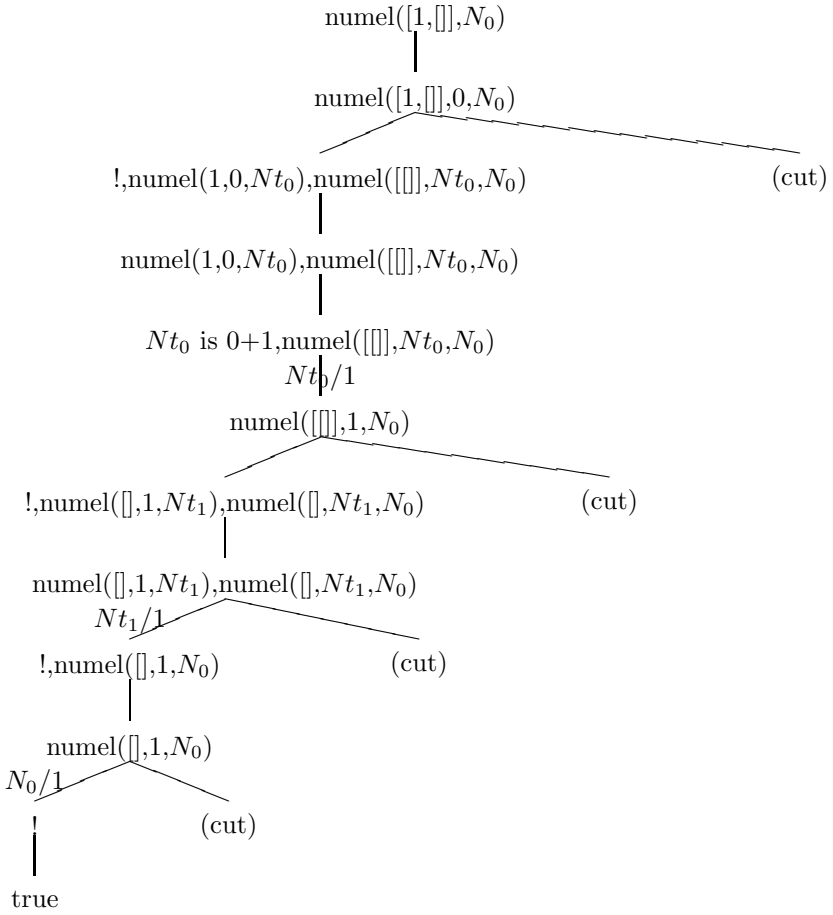
Min-Max



Alfa-beta



Esercizio 3



Esercizio 4

```
/*   no_dupl(Xs, Ys) is true if Ys is the list of the elements appearing  */
/*   in Xs without duplication.  The elements in Ys are in the same order  */
/*   as in Xs with the last duplicate values being kept.
*/
no_dupl([], []).
no_dupl([X|Xs], Ys):-
  member(X, Xs),
  no_dupl(Xs, Ys).
no_dupl([X|Xs], [X|Ys]):-
  nonmember(X, Xs),
  no_dupl(Xs, Ys).

/* member(X,Xs) is true if X is a member of the list
Xs.
*/
member(X,[X|Xs]).
member(X,[Y|Ys]):-member(X,Ys).

/* nonmember(X,Xs) is true if X is not a member of the list
Xs.             */
nonmember(_, []).
nonmember(X, [Y|Ys]):-X =\= Y, nonmember(X, Ys).
```

Esercizio 5

Si noti che esistono due sole configurazioni obiettivo:

C1 =
R B R
B R B
R B R

C2 =
B R B
R B R
B R B

Cioè le due possibili scacchiere (ottenibili una dall'altra invertendo i colori).

La funzione di valutazione del punto 2. invece, non poteva essere utilizzata come euristica ammissibile, per esempio in

B **R** B
R R R
B **R** B

vale 4 (numero delle coppie evidenziate), mentre la soluzione dista 1 (basta effettuare la mossa che inverte la casella centrale).

L'esercizio richiede di tracciare il comportamento dell'algoritmo a partire dalla configurazione data, in modo da poter giustificare ad ogni passo la scelta dello stato successivo.

R R R
B R R
B B B

B R R R B R R R B R R R R R R R R R R R R R R R
B R R B R R B R R R R R B B R B R B B R R B R R B R R
B B B B B B B B B B B B B B B B B B R B B B R B B B R

8 **5** 6 9 8 7 6 7 8

Lo stato successivo scelto corrisponde alla configurazione con valutazione migliore: la seconda da sinistra con valutazione 5. Iterando per altre 3 volte la generazione dei possibili stati successivi e la scelta dello stato migliore, si ottiene una soluzione ottima passando per stati con valutazione 3, 2 e 0.