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

12 Luglio 2007 (Tempo a disposizione 2h 1/2; su 32 punti)

Esercizio 1 (punti 7)
Si rappresentino in logica dei predicati del I ordine le seguenti frasi:

- Tutti i dipendenti della società "Acme" sono svogliati.
- Luca gioca a golf.
- Chiunque giochi a golf è svogliato.

Si rappresenti tale teoria in forma a clausole e si verifichi se tramite il principio di risoluzione con strategia *linear-input* se è possibile derivare una o più delle seguenti frasi (le si rappresenti in logica e poi a clausole):

- Tutti i dipendenti della società "Acme" giocano a golf
- Luca è una persona svogliata
- Tutte le persone svogliate giocano a golf
- Luca è un dipendente della società "Acme".

Esercizio 2 (punti 9)
Si consideri il seguente gioco. Si gioca in due ed ogni giocatore mostra con le dita di ogni mano un numero (da 0 a 4).
Ad ogni turno, un giocatore tocca con una delle sue mani una delle mani dell'avversario: al numero di questa viene sommato il numero della mano che l'ha toccata, dedotto di 5 se la somma eccede 4. Se una mano ha zero, va "fuori gioco" e non può essere né toccata dall'avversario né usata. Chi per primo ha zero in entrambe le mani perde.
Esempio (SD significa che il giocatore di turno tocca con la sua Sinistra la Destra dell'avversario, e così via). Partendo con tutti 1, dopo le mosse:
A: SS
B: SD
A: DS
B: DS
il giocatore A ha 2 nella mano S e 3 nella D, mentre B ha 0 in S e 1 in D.
Si mostri l'albero min-max, sviluppato fino al livello 3 (contando la radice come livello 0), supponendo che il primo giocatore a giocare sia *Max* e di partire dalla situazione in cui il giocatore *Max* ha 3 a destra e 3 a sinistra, mentre *min* ha 2 a sinistra e 1 a destra. Si eviti di sviluppare due figli dello stesso nodo se sono simmetrici. Si consideri come valore del nodo la differenza del numero di mani a 0 fra i due giocatori.
Si esplori l'albero con gli algoritmi min-max e alfa-beta.

Esercizio 3 (punti 5)
Si scriva un programma Prolog `sum(List,Num)` che data una lista `List` di numeri abbia successo se la somma di tali elementi è uguale a `Num`. Si noti che la lista può essere composta a sua volta da liste.
Esempio:

```
?- sum([3,2,1],X) yes X=6
?- sum([3,2,1,[], [[2,4], [1]]],X) yes X=13
```

Esercizio 4 (punti 7)

Il seguente programma Prolog serve per fare si` che due liste (costituite da variabili che possono assumere solo valori 1 e 2) abbiano esattamente *D* elementi diversi.

```
lds(L1,L2,0):-!,L1=L2.
lds([X|T],[X|TR],D):- lds(T,TR,D).
lds([X|T],[Rif|TR],D):- assegna(X), not(X=Rif), E is D-1,  lds(T,TR,E).
assegna(1).
assegna(2).
```

Si disegni poi l'albero SLDNF corrispondente al goal

```
?-lds([A,1,2,B,C],[1,2,2,2,1],1).
```

Esercizio 5 (punti 4)

Si spieghi brevemente il concetto di Consistenza di un grafo CSP e se ne illustrino i diversi gradi.

SOLUZIONE

Esercizio 1

Logica dei predicati:

- *Tutti i dipendenti della società "Acme" sono svogliati.*
- *Luca gioca a golf.*
- *Chiunque giochi a golf è svogliato.*

$\forall X$ dip(X, acme) $\Rightarrow$ svogliato(X)
gioca(luca,golf)
$\forall X$ gioca(X, golf) $\Rightarrow$ svogliato(X)

Query:

- *Tutti i dipendenti della società "Acme" giocano a golf*
- *Luca è svogliato*
- *Tutte le persone svogliate giocano a golf*
- *Luca è un dipendente della società "Acme".*

$\forall X$ dip(X, acme) $\Rightarrow$ gioca(X,golf)
svogliato(luca)
$\forall X$ svogliato(X) $\Rightarrow$ gioca(X, golf)
dip(luca,acme)

Clausole (definite):

1) not dip(X, acme) or svogliato(X)
2) gioca(luca,golf)
3) not gioca(X, golf) or svogliato(X)

Clausole goal (Horn):

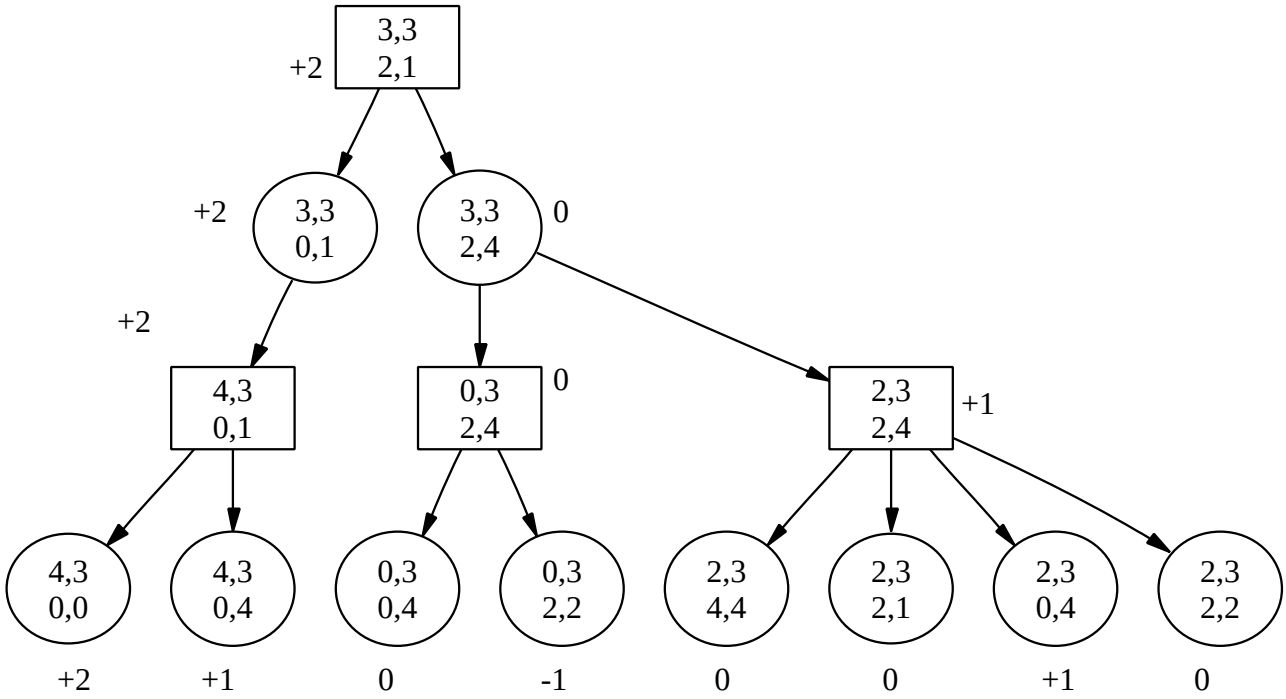
4) G1: not ($\forall X$ dip(X, acme) $\Rightarrow$ gioca(X,golf))
   $\exists X$ not (not dip(X, acme) or gioca(X,golf))
   $\exists X$ dip(X, acme) and not gioca(X,golf) Skolemizzazione:
   dip(c, acme) and not gioca(c,golf)
   G1') dip(c, acme)
   G1'') not gioca(c,golf)
5) G2: not svogliato(luca)
6) G3: not ($\forall X$ svogliato(X) $\Rightarrow$ gioca(X,golf))
   $\exists X$ not (not svogliato(X) or gioca(X,golf))
   $\exists X$ svogliato(X) and not gioca(X,golf) Skolemizzazione:
   svogliato(d) and not gioca(d,golf)
   G3') svogliato(d)
   G3'') not gioca(d,golf)
7) G4: not dip(luca,acme)

Risoluzione:

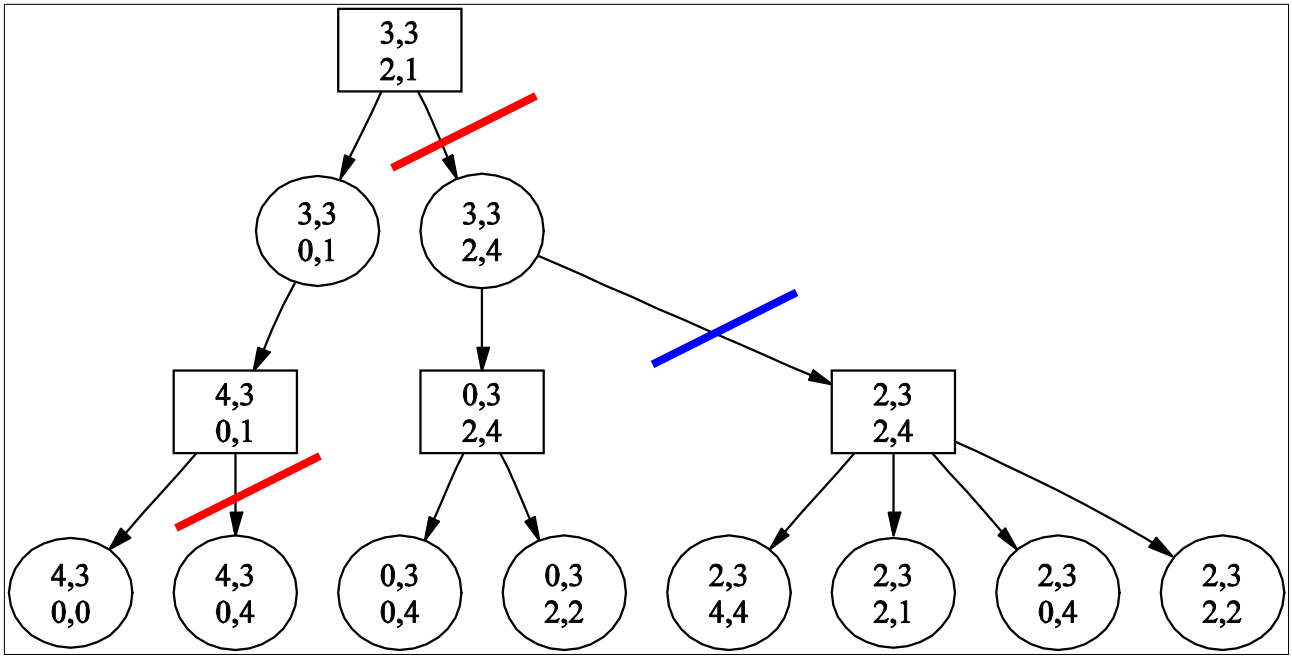
8) da G1'+1): svogliato(c) - nessun altro passo
9) da G2+3+2): clausola vuota in due passi
10) da G3'+ ... nessun passo di risoluzione
11) da G4+ ... nessun passo di risoluzione

Esercizio 2

min-max:



Alfa-beta:



+2

Nota: in rosso i tagli effettuati considerando che il valore +2 è massimo per MAX, e rappresenta la vittoria al gioco; in blu il taglio effettuato qualora non fosse svolta questa considerazione.

Esercizio 3

```
sum(List,Num)

sum([],0).
sum ([X|Y],N):- is_list(X), !,sum(X,N1), sum(Y,N2), N is N1 + N2.
sum ([X|Y],N):-sum(Y,N1), N is N1 + X.

is_list([]).
is_list([_|_]).
```

Esercizio 4: