FONDAMENTI DI INTELLIGENZA ARTIFICIALE M

15 Luglio 2010 – Tempo a disposizione 2h 45min – Risultato 32/32 punti

Esercizio 1 (punti 4)

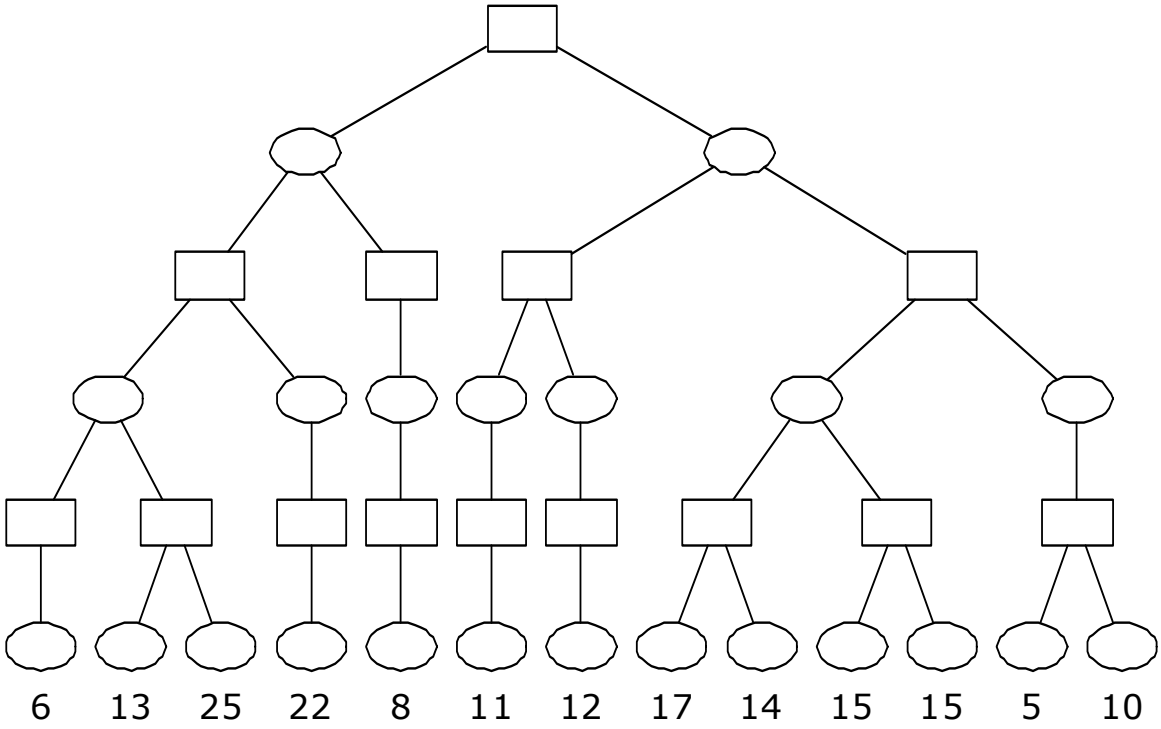
Si formalizzino il logica dei predicati del I ordine le seguenti frasi:

- *Ogni bimbo ama ogni caramella*
- *Chiunque ami una caramella non è un nutrizionista*
- *Chiunque mangi qualche zucca è un nutrizionista*
- *Chiunque acquisti qualsiasi zucca o la intaglia o la mangia (or esclusivo)*
- *Giovanni acquista una zucca*
- *Giovanni è un bimbo*
- *Golia è una caramella*

Dimostrare poi applicando il principio di risoluzione che: *Giovanni intaglia una zucca*

Esercizio 2 (punti 4)

Si consideri il seguente albero di gioco in cui la valutazione dei nodi terminali è dal punto di vista del primo giocatore. Si assuma che il primo giocatore sia Max.



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

Esercizio 3 (punti 4)

Il seguente programma Prolog verifica se in una lista tutti gli elementi sono diversi:

```
alldiffer(L):- not(twoequal(L)).
twoequal(L):- membrest(X,T,L), membchk(X,T), !.
membrest(X,T,[X|T]).
membrest(X,T,[_|L]):- membrest(X,T,L).
```

Si mostri l'albero di derivazione SLDNF relativo al goal

```
:- alldiffer([p(r,a),p(A,r),p(a,R)]).
```

Si noti che il predicato membchk/2 è da intendersi come il predicato member/2 usuale.

Esercizio 4 (punti 4)

Nel seguente problema si tratta di trovare una sistemazione delle cifre da 1 a 8 nella scacchiera raffigurata sotto, in modo tale che ogni cifra non sia consecutiva a nessuna delle cifre in caselle adiacenti (in orizzontale, verticale o diagonale). Ad esempio la configurazione in figura non sarebbe una soluzione, perché 1 e 2 sono in caselle adiacenti, come anche 2 e 3, 7 e 8, 5 e 6.

Si formuli il problema come un problema di soddisfacimento di vincoli e lo si risolva tramite il *forward checking*.

	1	
2	7	4
3	8	6
	5	

Nel mostrare come si ottiene la soluzione, si scelga, ad ogni passo, come variabile da istanziare quella che ha il valore più piccolo nel dominio (e in caso di parità, si istanzi la variabile soggetta a più vincoli. Se ancora si presentano casi di parità in base, si selezioni prima la variabile con indice più basso.

Come euristica di selezione del valore, si assegnino i valori in ordine crescente.

Esercizio 5 (punti 3)

Dato un insieme di studenti e i voti ottenuti negli esami, rappresentati come fatti del tipo:
`studente(Nome, Voto).`
si definisca il predicato `studMedia(Nome,Media)` che dato il nome di uno studente (`Nome`) ne determina la media in `Media`.

Esercizio 6 (punti 2)

Si dia la definizione di *euristica ammissibile* e *euristica monotona*. La monotonicità della *f* euristica ne garantisce l'ammissibilità? Perché sono importanti le euristiche ammissibili? E quelle monotone?

Esercizio 7 (punti 5)

Si scriva un meta interprete per Prolog che per la selezione delle clausole non segua l'ordine testuale, ma selezioni sempre prima quella che ha un numero minore di sottogoals (a partire dai fatti).
Ad esempio, nel caso delle seguenti clausole:

```
1. p(X,Y,Z):- b(Y, Y),a(X, Y), c(Z).
2. p(X,Y,Z).
3. p(X,Y,Z):- s(Y, Y), s(Y,Y).
4. p(X,Y,Z):- s(Y, Y).
```

L'ordine di selezione sarà 2. 4. 3. e 1.
Si supponga di avere a disposizione un predicato `sort(L, L1, L2)` che fornisce in L2 la lista L ordinata (in senso crescente) secondo i valori contenuti in L1.
Si supponga, inoltre, di avere un programma nella forma `clausola(Head, Body)` dove `Body` è una lista di sottogoal. I fatti hanno come Body la lista vuota.

Esercizio 8 (punti 3)

Si consideri il seguente problema di pianificazione. Ci sia un robot in grado di muoversi, di afferrare e di depositare oggetti all'interno di un ambiente composto da due stanze comunicanti *S1* e *S2*. Lo stato iniziale, l'obiettivo, le precondizioni e gli effetti di ogni azione sono formulati in STRIPS:

STATO INIZIALE (At(Robot,S1) ∧ At(Object2,S1) ∧ At(Object1,S2) ∧ Room(S1) ∧ Room(S2) ∧ HandEmpty)
GOAL (At(Object1,S1) ∧ At(Object2,S1))
ACTION (Go(x,y),
PRECOND: Room(x) ∧ Room(y) ∧ At(Robot,x) ∧ x≠y
EFFECT: ¬ At(Robot,x) ∧ At(Robot,y))
ACTION (Pick(o),
PRECOND: Room(x) ∧ At(Robot,x) ∧ At(o,x) ∧ HandEmpty ∧ o≠Robot
EFFECT: ¬HandEmpty ∧ ¬At(o,x) ∧ Holding(o))
ACTION (Drop(o),
PRECOND: Room(x) ∧ At(Robot,x) ∧ Holding(o)
EFFECT: HandEmpty ∧ At(o,x) ∧ ¬Holding(o))

Le variabili iniziano con lettera minuscola, le costanti conn lettera maiuscola. Si risolva il problema di pianificazione con una **pianificazione in avanti** mediante una ricerca **A* con eliminazione degli stati ripetuti** e considerando come funzione euristica il numero di letterali di goal *non* soddisfatti nello stato.

Nella costruzione dell'albero, si applichino le azioni nell'ordine dato sopra. A parità di altro, si scelga il nodo generato prima. Si rappresentino gli stati in modo grafico. Si riporti l'albero di ricerca generato (indicando, per ogni nodo, il valore della funzione di valutazione e l'ordine di espansione) e la soluzione trovata.

Si noti che l'azione Pick(o) ha come conseguenza che per l'oggetto 'o' verrà a valere ¬At(o,x).

Esercizio 9 (punti 1)

Si spieghi in cosa consiste l'anomalia di Sussmann con un esempio.

Esercizio 10 (punti 2)

Descrivere brevemente in cosa consiste la Open World Assumption (OWA) e la Close World Assumption (CWA), fornendo un breve esempio di base di conoscenza che mostri come tali assunzioni modificano il ragionamento su tale base di conoscenza.

SOLUZIONE

Esercizio 1

1. $\forall X \forall Y$ (bimbo(X) and caramella(Y) $\rightarrow$ ama(X,Y))
2. $\forall X \forall Y$ (caramella(Y) and ama(X,Y) $\rightarrow \neg$ nutriz(X))
3. $\forall X ((\exists Y$ (zucca(Y) and mangia(X,Y))) $\rightarrow$ nutriz(X))
4. $\forall X \forall Y$ (zucca(Y) and compra(X,Y) $\rightarrow$ intaglia(X,Y) xor mangia(X,Y))
5. $\exists Y$ (zucca(Y) and compra(giovanni,Y))
6. bimbo(giovanni)
7. caramella(golia)
8. $\neg (\exists Y$ (zucca(Y) and intaglia(giovanni,Y)))
   equivale a : $\forall Y$ (zucca(Y) and intaglia(giovanni,Y))

Clausole:

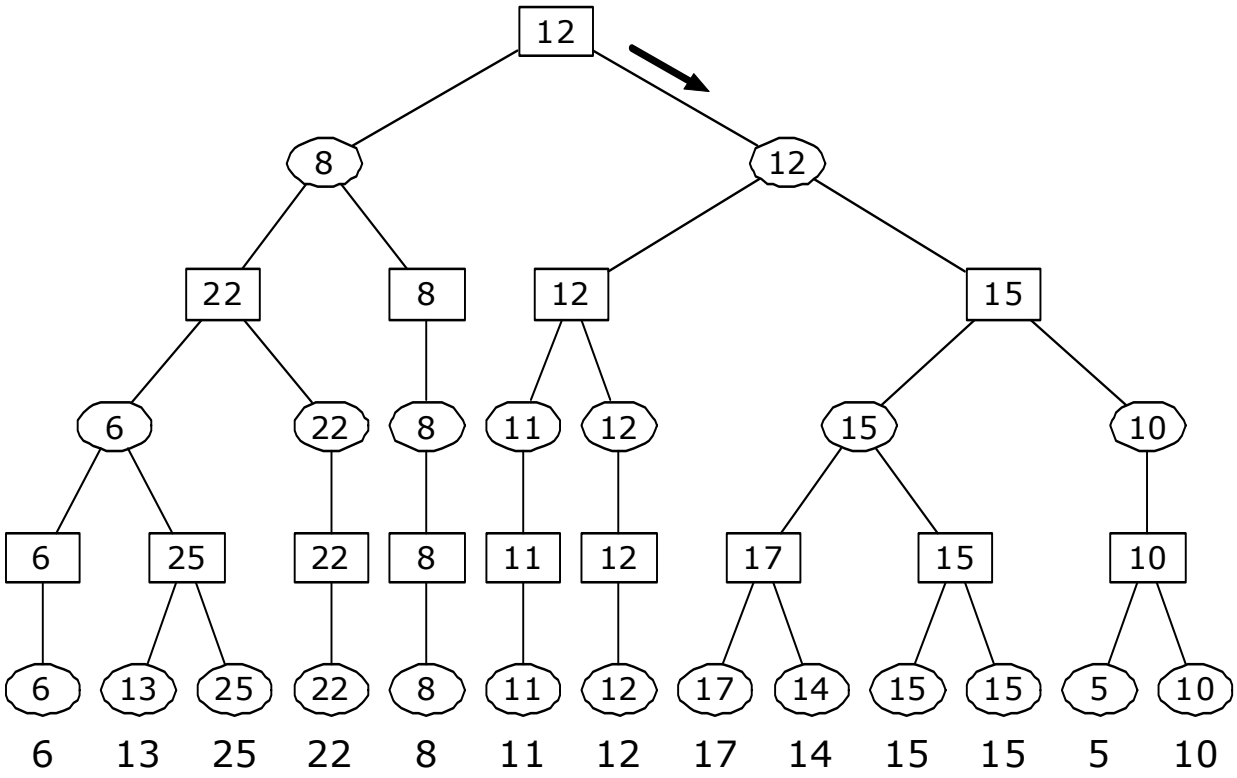
1 ¬bimbo(X) or ¬caramella(Y) or ama(X,Y)
2 ¬caramella(Y) or ¬ama(X,Y) or ¬ nutriz(X)
3 ¬zucca(Y) or ¬ mangia(X,Y) or nutriz(X)
4 ¬zucca(Y) or ¬compra(X,Y) or intaglia(X,Y) or mangia(X,Y)
5 ¬zucca(Y) or ¬compra(X,Y) or ¬intaglia(X,Y) or ¬mangia(X,Y)
6 zucca(c)
7 compra(giovanni,c)
8 bimbo(giovanni)
9 caramella(golia)
10 ¬ zucca(Y) or ¬intaglia(giovanni,Y)

Risoluzione:

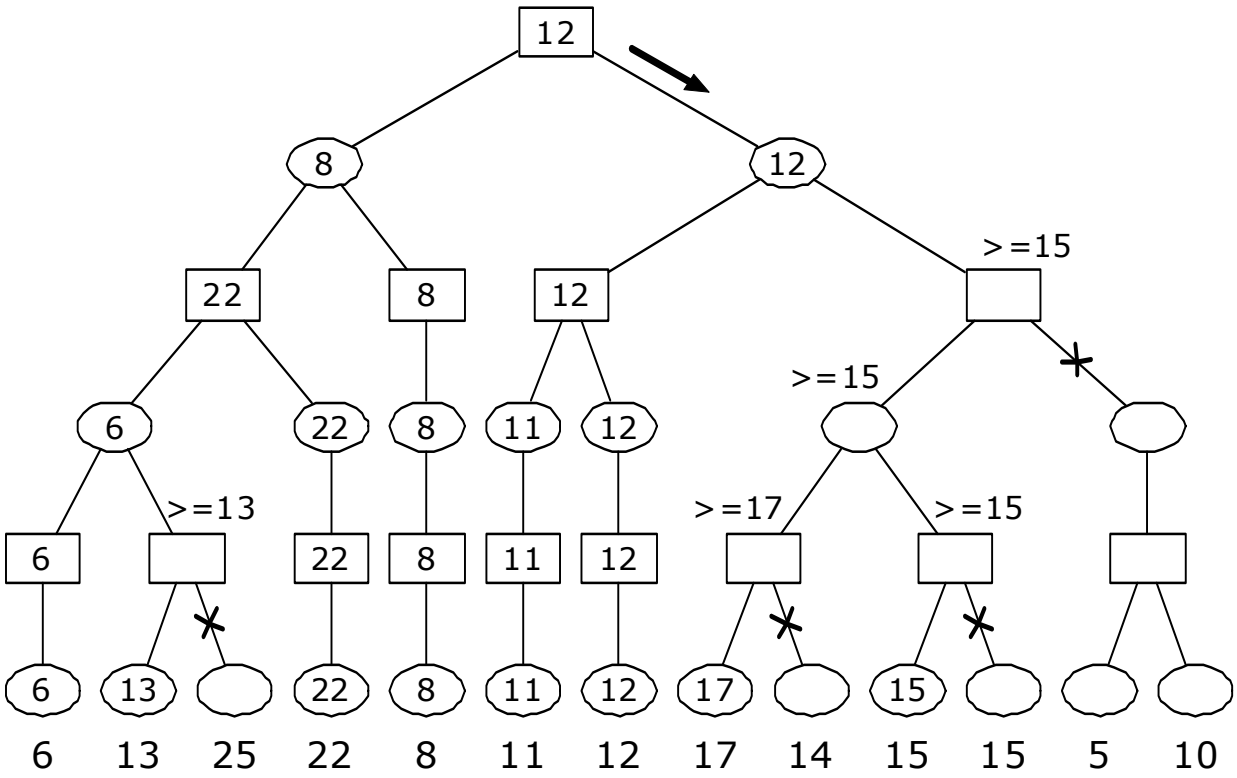
11. ama(giovanni,golia) da 8+9+1
12. ¬ nutriz(giovanni) da 11+9+2
13. intaglia(giovanni,c) or mangia(giovanni,c) da 6+7+4
14. mangia(giovanni,c) da 6+10+13
15. nutriz(giovanni) da 3+13+6
16. vuota da 15+12

Esercizio 2

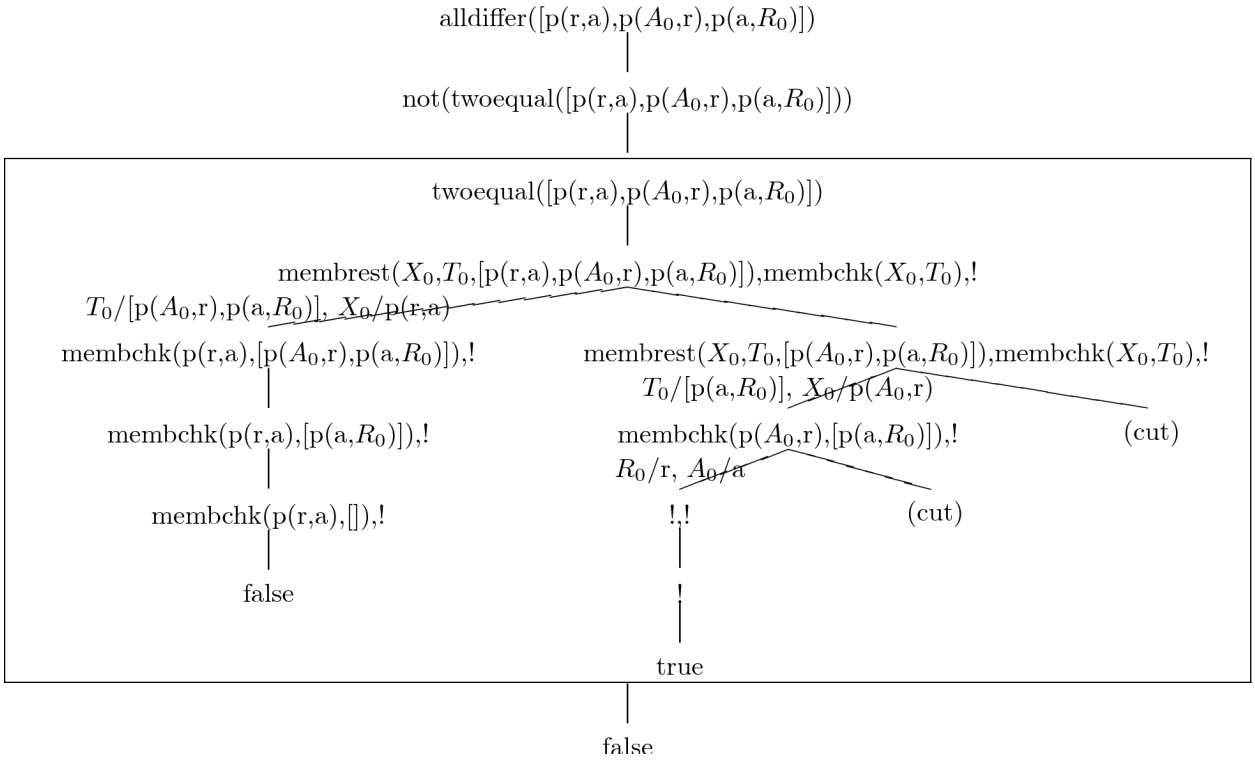
min-max:



Alfa-beta:



Esercizio 3



Esercizio 4

	X1	
X2	**X3**	**X4**
X5	**X6**	**X7**
	X8	

Formulazione CSP:

- *variabili*: X1, … , X8, associate alle caselle come in figura
- *domini*: valori da 1 a 8
- *vincoli*:
  - ∀i,j, i≠j ➔ Xi≠Xj
  - adiacente(i,j) ➔ Xi≠Xj+1, Xi≠Xj-1
  - adiacente(1,2), adiacente(1,3), adiacente(1,4), adiacente(2,3), adiacente(2,5), adiacente(2,6), adiacente(3,4), adiacente(3,5), adiacente(3,6), adiacente(3,7), adiacente(4,5), adiacente(4,7), adiacente(5,6), adiacente(5,8), adiacente(6,7), adiacente(6,8), adiacente(7,8).

Forward Checking:
Tutte le variabili hanno inizialmente lo stesso dominio.
Le variabili soggette a più vincoli sono X3 e X6; partiamo da X3.
X3=1

X1	X2	X3	X4	X5	X6	X7	X8
3..8	3..8	**1**	3..8	3..8	3..8	3..8	2..8

Ora la variabile che ha il valore più piccolo nel dominio e` X8 (che ha il valore 2):

X1	X2	X3	X4	X5	X6	X7	X8
3..8	3..8	**1**	3..8	4..8	4..8	4..8	**2**

Il valore più piccolo nei domini e` 3 ed e` nel dominio delle variabili X1, X2, X4. X2 e X4 hanno più vincoli di X1, in quanto X1 ha vincoli solo con X2 e X4, mentre X2 ha vincoli con X5 e X6 e X4 ha vincoli con X6 e X7 (oltre al vincolo di diverso fra tutte le variabili, che e` comune a tutte). Istanziamo quindi X2:

X1	X2	X3	X4	X5	X6	X7	X8
5..8	**3**	**1**	4..8	5..8	5..8	4..8	**2**

Ora il valore piu` piccolo e` 4 ed e` nei domini di X4 e X7. X4 ha vincoli con 3 variabili (X1, X6 e X7), mentre X7 solo con 2 (X4 e X6). Istanziamo quindi X4:

X1	X2	X3	X4	X5	X6	X7	X8
6..8	**3**	**1**	**4**	5..8	6..8	6..8	**2**

Il valore più piccolo e` nel dominio di X5:

X1	X2	X3	X4	X5	X6	X7	X8
6..8	**3**	**1**	**4**	**5**	7..8	6..8	**2**

Il valore più piccolo e` nel dominio di X1 e X7; X7 ha un vincolo con X6 che X1 non ha:

X1	X2	X3	X4	X5	X6	X7	X8
7..8	**3**	**1**	**4**	**5**	8	**6**	**2**

X1 ha l'elemento più piccolo nel dominio e viene istanziato a 7. Infine si istanza X6 a 8.
La soluzione e`:

	7	
3	**1**	**4**
5	**8**	**6**
	2	

Esercizio 5

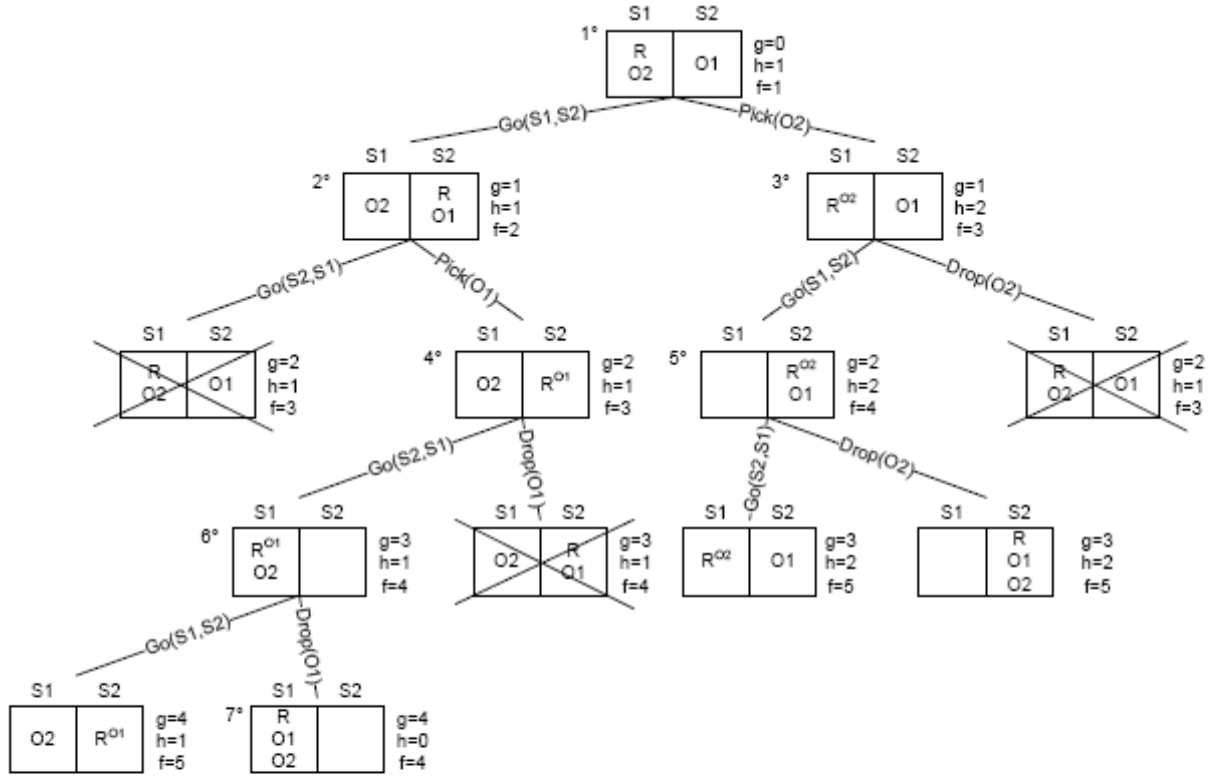
```
studMedia(Nome,Media):- findall(Voto,studente(Nome,Voto),List),
                        media(List, Media).


%semplice predicato per calcolare la media su una lista di numeri
media(Lista,Media) :- length(Lista,LunghezzaLista),
                      sum(Lista,Somma),
                      Media is Somma / LunghezzaLista.
```

Esercizio 7

```
sum([],0).
sum([A|B],N):- sum(B,N1),N is N1+1.

solve([]).
solve([Head|Tail]):-
     solve(Head),
```

```
     solve(Tail).

solve(Goal):-
     findall((Goal,Body),clausola(Goal,Body),
                                                    L),
     ordina(L,Lord),
     member((Goal,FirstBody), Lord),
     solve(FirstBody).

ordina(L,Lord):-
     conta_sottogoal(L,Nsottogoal),
     sort(L,Nsottogoal, Lord).

conta_sottogoal([],[]).
conta_sottogoal([H|T],[NH|NT]):-
     conta_sottogoal1(H,NH),
     conta_sottogoal(T,NT).

conta_sottogoal1((_,Body),Nsottogoal):-conta(Body, Nsottogoal)

conta([], 0):- !.
conta([Arg|Tail],  Nout):-
     conta(Tail, Nin), Nout is Nin +1.
```

Esercizio 8



La soluzione trovata è Go(S1,S2), Pick(Object1), Go(S2,S1), Drop(Object1).