COMPITO DI INTELLIGENZA ARTIFICIALE (v.o.) – PARTE I
FONDAMENTI DI INTELLIGENZA ARTIFICIALE
21 Giugno 2005 (Tempo a disposizione 2,30 h; su 32 punti)

Esercizio 1: (punti 6)
Si traducano in logica dei predicati del primo ordine le seguenti frasi:

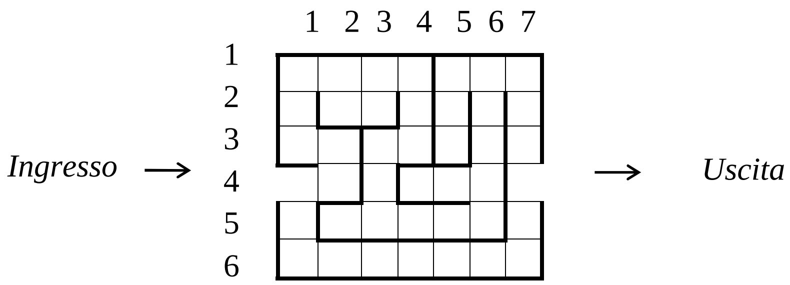
A. *Se qualcuno risparmia del denaro guadagna un certo interesse.*
B. *Se non ci fosse un interesse nessuno risparmierebbe denaro.*

Dimostrare con il metodo di risoluzione che *B* è conseguenza logica di *A*. Si utilizzi nella formalizzazione il seguente vocabolario: rd(X, Y): X risparmia la somma di denaro Y
gi(X, Z): X guadagna l'interesse Z

Esercizio 2: (punti 6)
Si scriva un programma PROLOG che data in ingresso una lista di liste Lin, dia in uscita una nuova lista di liste Lout in cui ogni lista interna non condivide elementi con la precedente lista interna. Si supponga per semplicità che le liste interne non possiedano elementi ripetuti. Si definiscano tutti i predicati utilizzati.

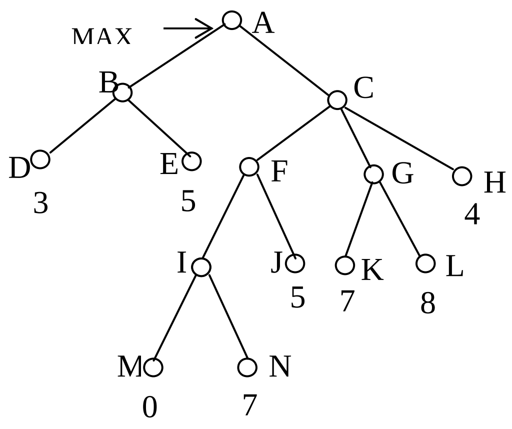
Esempio:
```
?-reduce([[1,5,2,4,9], [3,7,2], [9,3]],Z).
Z = [[1,5,2,4,9], [3,7], [9]]
```

Esercizio 3: (punti 10)
Dato il seguente labirinto rappresentato su una griglia 6x7:



a. formalizzare il problema di trovare un percorso dall'ingresso all'uscita come un problema di ricerca in uno spazio di stati;
b. descrivere un algoritmo di tipo A* per i grafi per trovare la soluzione; *Suggerimento: si scelga come stima la Manhattan distance tra due punti (x1,y1), (x2,y2) che è data dalla formula:* $M = |y2 - y1| + |x2 - x1|$
c. mostrare come si comporta l'algoritmo (mediante l'albero di ricerca). Trova la soluzione ottimale?

Esercizio 4: (punti 7)
Sia dato il seguente esempio di gioco con avversario in cui si assume una funzione di valutazione degli stati terminali che valuta le foglie come indicato in figura:



a) Evidenziare le valutazioni dei nodi mediante l'algoritmo MIN MAX e dire quale mossa farebbe MAX.
b) Tracciare l'esecuzione dell'algoritmo con potatura α-β con visita dell'albero da sinistra a destra (evidenziando le potature). Tracciare l'esecuzione di α-β con visita dell'albero da destra a sinistra. La mossa scelta è la stessa nei due casi? I nodi non visitati sono gli stessi nei due casi? Perché?

Esercizio 5: (punti 3)
Si definisca che cosa si intende per un sistema di inferenza corretto e completo. Si citi un esempio di sistema di inferenza rispettivamente:

- corretto e completo;
- non-corretto.

Soluzione:

Esercizio 1:

Si utilizza nella formalizzazione il seguente vocabolario:

rd(X, Y): X risparmia la somma di denaro Y

gi(X, Z): X guadagna l'interesse Z

A. $\forall X\ (\exists Y\ rd(X, Y)) \Rightarrow (\exists Z\ gi(X, Z))$

B. $\forall X\ \neg(\exists Z\ gi(X, Z)) \Rightarrow \neg(\exists Y\ rd(X, Y))$

Trasformazione in FC di A	Trasformazione in FC di B negato
$\forall X(\neg\ (\exists Y\ rd(X, Y)) \vee (\exists Z\ gi(X, Z)))$	$\neg\forall X\ (\exists Z\ gi(X, Z) \vee (\neg\exists Y\ rd(X, Y))$
$\forall X\ ((\forall Y\ \neg\ rd(X, Y)) \vee (\exists Z\ gi(X, Z)))$	$\exists X\ \neg((\exists Z gi(X, Z)) \vee (\neg\exists Y\ rd(X, Y)))$
$\forall X\ ((\forall Y\ \neg\ rd(X, Y)) \vee gi(X, l(X)))$	$\exists X\ (\neg\exists Z\ gi(X, Z) \wedge \exists Y\ rd(X, Y))$
$\{\neg\ rd(X, Y), gi(X, l(X))\}$	$\exists X\ (\forall Z\ \neg gi(X, Z) \wedge \exists Y\ rd(X, Y))$
	$\forall Z\ \neg gi(s1, Z) \wedge \exists Y\ rd(s1, Y)$ s1 costante
	$\forall Z\ \neg gi(s1, Z)) \wedge rd(s1, s2)$ s2 costante
	$\{\neg gi(s1, Z)\}$
	$\{ rd(s1, s2)\}$

Le formalizzazioni alternative

A'. $\forall X\ \forall Y\ (rd(X, Y) \Rightarrow (\exists Z\ gi(X, Z)))$

B'. $\forall X\ (\forall Z \neg gi(X, Z)) \Rightarrow \neg(\exists y\ rd(X, Y)))$

A''. $\exists Z \forall X\ \forall Y\ (rd(X, Y) \Rightarrow (gi(X, Z)))$

sono anche plausibili e consentono di svolgere correttamente l'esercizio.

Le seguenti invece non lo sono:

A'. $\forall X\ (\forall y\ rd(X, Y)) \Rightarrow (\exists Z\ gi(X, Z))$

per guadagnare un interesse è necessario risparmiare tutte le possibili somme di denaro

A'. $\forall X\ \forall Y\ \forall Z\ rd(X, Y) \Rightarrow gi(X, Z)$

Data una qualunque persona x, somma risparmiata y e interesse z è vero che se x risparmia y ottiene un interesse z.

B'. $\forall X\ \forall Z\ (\neg gi(X, Z) \Rightarrow \neg(\exists Y\ rd(X, Y))$

Se per una qualunque persona e qualunque interesse è il caso che la persona non guadagna quell'interesse allora la persona non risparmia denaro. Per esempio dal fatto che non guadagno un interesse di qualche miliardo posso concludere che non è interessante risparmiare.

Grafo di refutazione:

{¬ rd(X, Y), gi(X, l(X))}　　　{ rd(s1, s2)}　　　{¬gi(s1, Z)}

{X/s1, Y/s2}

{gi(s1, l(s1)}

{Z/l(s1)}

{}

Esercizio 2:

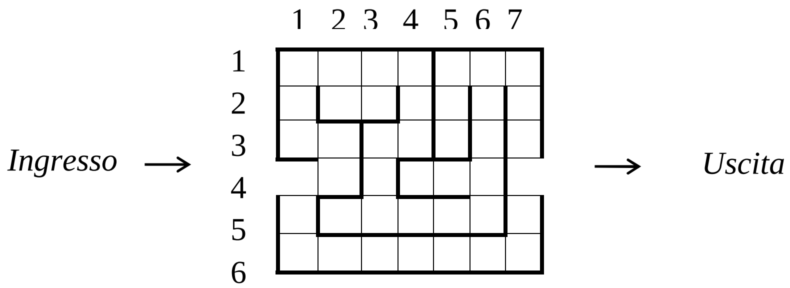
```
reduce([],[]).
reduce([A],[A]):-!.
reduce([A,B|C],[A|S]):-test(A,B,T),reduce([T|C],S).

test([],S,S).
test([A|B],S,T):-delete(A,S,T1),test(B,T1,T).

delete(El,[],[]):-!.
delete(A,[A|B],B):-!.
delete(A,[C|T],[C|B]):- delete(A,T,B).
```

Esercizio 3:

Possiamo rappresentare il labirinto su una griglia:



Indichiamo una posizione nel labirinto con una coppia di coordinate (riga, colonna).

Stati: una coppia di coordinate (x,y) appartenenti al labirinto
Stato iniziale: Ingresso in (4,1)
Goal State: Uscita in (4,7)
Operatori: posso andare su, giù a destra o a sinistra dove non c'è il muro.
Ad esempio, dallo stato (4,1) possiamo andare agli stati (5,1), (4,2)

Algoritmo A*: è un algoritmo Best First con euristica h ammissibile e cioè h(n) è una sottostima rispetto alla valutazione h*(n).

$f(n) = h(n) + g(n)$

dove h(n) stima del costo per raggiungere l'uscita, g(n) costo del cammino percorso. Possiamo prendere come h(n) la Manhattan distance tra lo stato n e lo stato goal e come g(n) il numero di caselle che l'agente ha percorso.Potremmo considerare come h(n) anche la distanza in linea d'aria tra lo stato n e l'uscita, ma la Manhattan distance domina la distanza in linea d'aria pur rimanendo una sottostima. Preferiamo quindi scegliere questa come stima. La Manhattan distance tra due punti (x1,y1), (x2,y2) è data dalla formula

$M = |y2 - y1| + |x2 - x1|$

Sull’esempio con goal state (4,7) l’algoritmo A* genera il seguente albero di ricerca

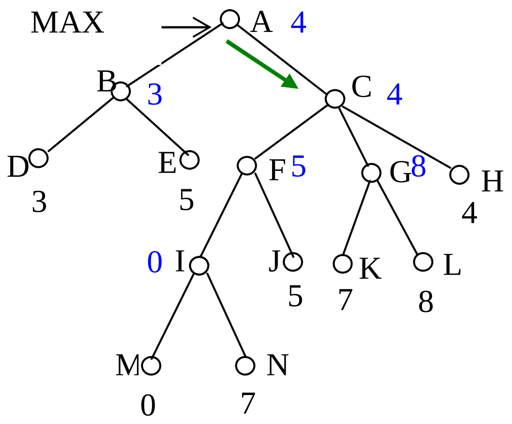
(4,1)

(5,1)	(4,2)
f = 6+1+1=8	f = 5+1=6
(6,1)	(3,2)
f=6+2+2=10	f = 5+1+2=8
(6,2)	(3,1)
f=5+2+3=10	f = 6+1+3=10
(6,3)	(2,1)
f=4+2+4=10	f = 6+2+4=12
(6,4)	
f=3+2+5=10	
(6,5)	
f=2+2+6=10	
(6,6)	
f=1+2+7=10	
(6,7)	
f=0+2+8=10	
(5,7)	
f=0+1+9=10	
(4,7)	
goal!	

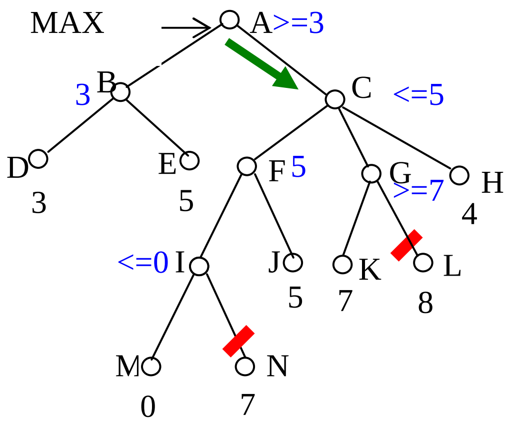
La soluzione trovata è ottimale perché è il percorso minimo tra l’ingresso e l’uscita del labirinto.

Esercizio 4:

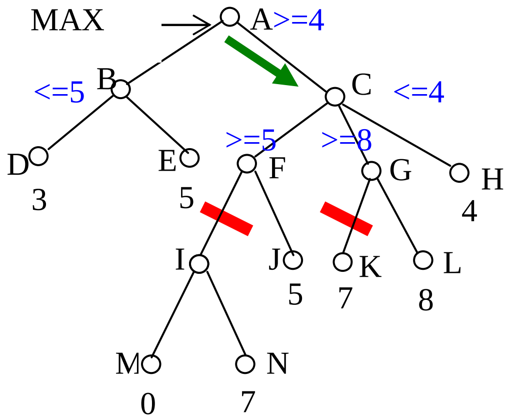
min-max



Alfa-beta da sinistra a destra



Alfa-beta da destra a sinistra



La mossa scelta dall’algoritmo alfa-beta (nelle due visite) e dal min-max risulta la stessa. Questo non è sempre vero: qualora ci siano mosse equivalenti (che portano allo stesso valore di euristica) con visite diverse si potrebbero ottenere scelte diverse. Il valore di euristica associato al nodo scelto sarebbe comunque lo stesso.

Il numero di nodi visitati dipende dall’ordine di visita: se vengono visitati prima i nodi migliori, il numero di nodi visitati dall’algoritmo alfa-beta diminuisce.