

Strategie di ricerca

Esercizi – Martedì 20 Giugno 2006

- **Scopo:**
 1. Esercizi sulle strategie di ricerca
 2. Esercizi sui giochi, alberi min-max e tagli alfa-beta
 3. Qualche esercizio su Prolog

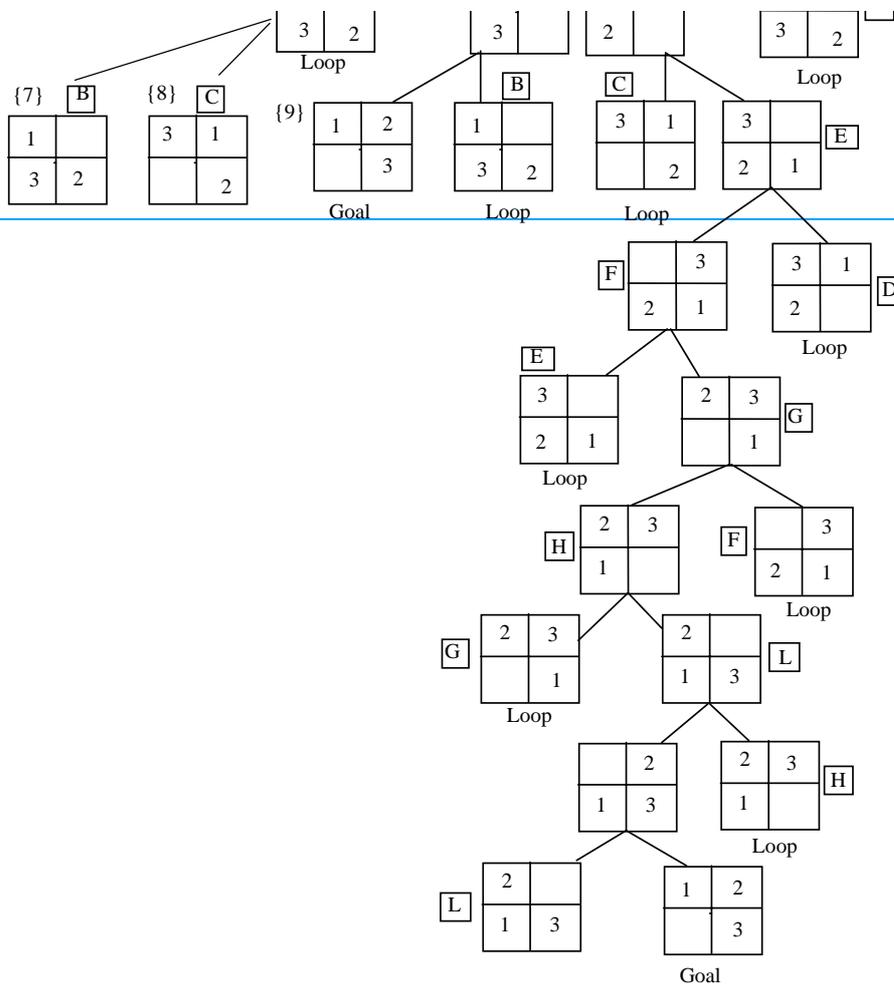
1

Ricordiamoci che...

- Una funzione euristica $h(n)$ è detta **ammissibile** se **non sbaglia mai per eccesso** la stima del costo per arrivare all'obiettivo
- Una funzione euristica $h(n)$ è detta **consistente** (o **monotona**) se, per ogni nodo n ed ogni successore n' generato da n applicando l'azione a , vale:

$$h(n) \leq c(n, a, n') + h(n')$$

2

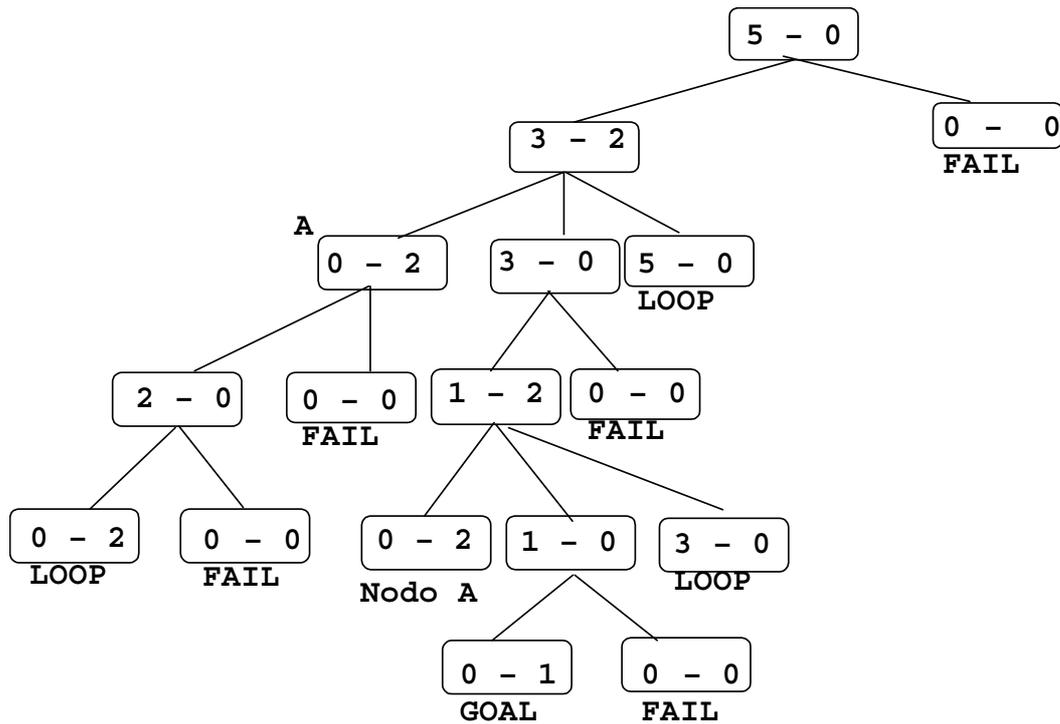


Esercizio – strategie di ricerca

Si consideri il seguente problema. Si hanno due contenitori per liquidi, uno da cinque litri completamente pieno d'acqua e uno da due litri vuoto.

- Vogliamo ottenere precisamente un litro d'acqua nel contenitore da due litri.
- Possiamo trasferire acqua da un contenitore all'altro, o buttarla, ma non ottenerne di nuova.
- Si mostri tutto lo spazio degli stati per risolvere il problema. Si indichi, inoltre, dopo quanti passi si giungerebbe alla soluzione se si utilizzasse una strategia breadth-first e depth-first.

Soluzione – strategie di ricerca



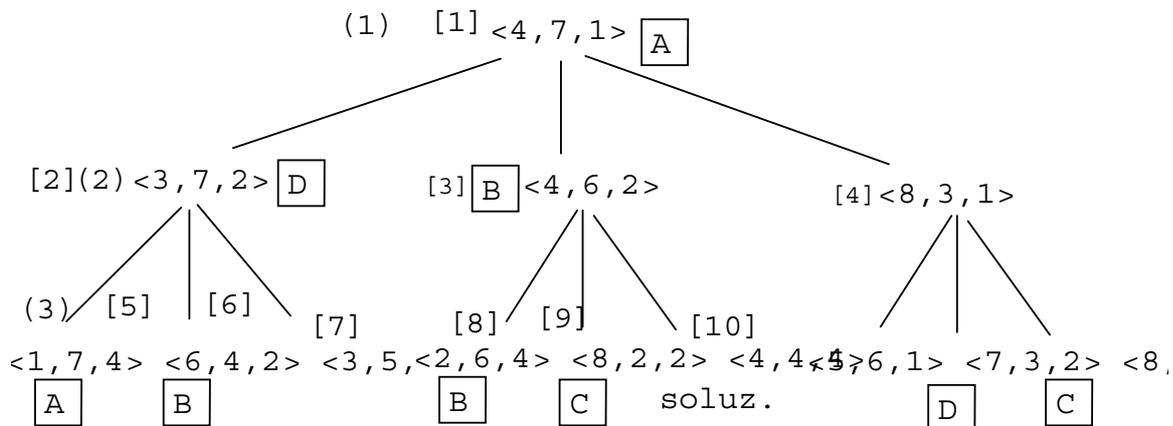
7

Esercizio – strategie di ricerca

- Si supponga di avere 12 monete ripartite in tre distinte pile. La prima pila è formata da 4 monete, la seconda da 7 e la terza da 1. Lo spostamento delle monete deve avvenire secondo la seguente regola: ogni volta che si spostano delle monete da una pila A ad un'altra B, le monete in B devono raddoppiare di numero.
- Si mostri lo spazio di ricerca con stato iniziale rappresentato dalla tripla $\langle 4, 7, 1 \rangle$ e stato finale dalla tripla $\langle 4, 4, 4 \rangle$ generato in due spostamenti e lo si esplori con strategia in ampiezza e profondità.

8

Soluzione – strategie di ricerca



9

Esercizio – strategie di ricerca

- Si consideri un puzzle con la seguente configurazione iniziale:

N	N	N	B	B	B	V
---	---	---	---	---	---	---

- Ci sono tre tessere Nere (N) e tre tessere bianche (B) e una cella vuota (V).
- Il puzzle ha le mosse seguenti:
 - (a) Una tessera si può spostare in una adiacente vuota con costo unitario.
 - (b) Una cella si può spostare ed inserirsi nella cella vuota saltando al più due altre tessere con costo uguale alla distanza coperta (al numero delle tessere saltate più 1).

10

Esercizio – strategie di ricerca

- Il goal del puzzle è avere tutte le tessere bianche a sinistra di quelle nere, indipendentemente da dove sia la tessera vuota.
- Si costruisca una funzione euristica $f = g + h'$ per questo problema dove g è stato specificato precedentemente e si mostri l'albero di ricerca prodotto dall'algoritmo A^* .

11

Esercizio – strategie di ricerca

- Esempi di EURISTICHE:
- 1. Si può considerare il numero di tessere fuori posto: le tessere nere sono fuori posto se occupano le caselle 1, 2, 3 mentre le bianche sono fuori posto se occupano le celle 5, 6, 7.

1	2	3	4	5	6	7
N	N	N	B	B	B	V

- $h'=5$

12

Esercizio – strategie di ricerca

- Esempi di EURISTICHE:
- 2. Come euristica si può considerare la distanza che separa le tessere nere dalle posizioni 4, 5, 6.
- Euristica non ammissibile: infatti, nella configurazione goal

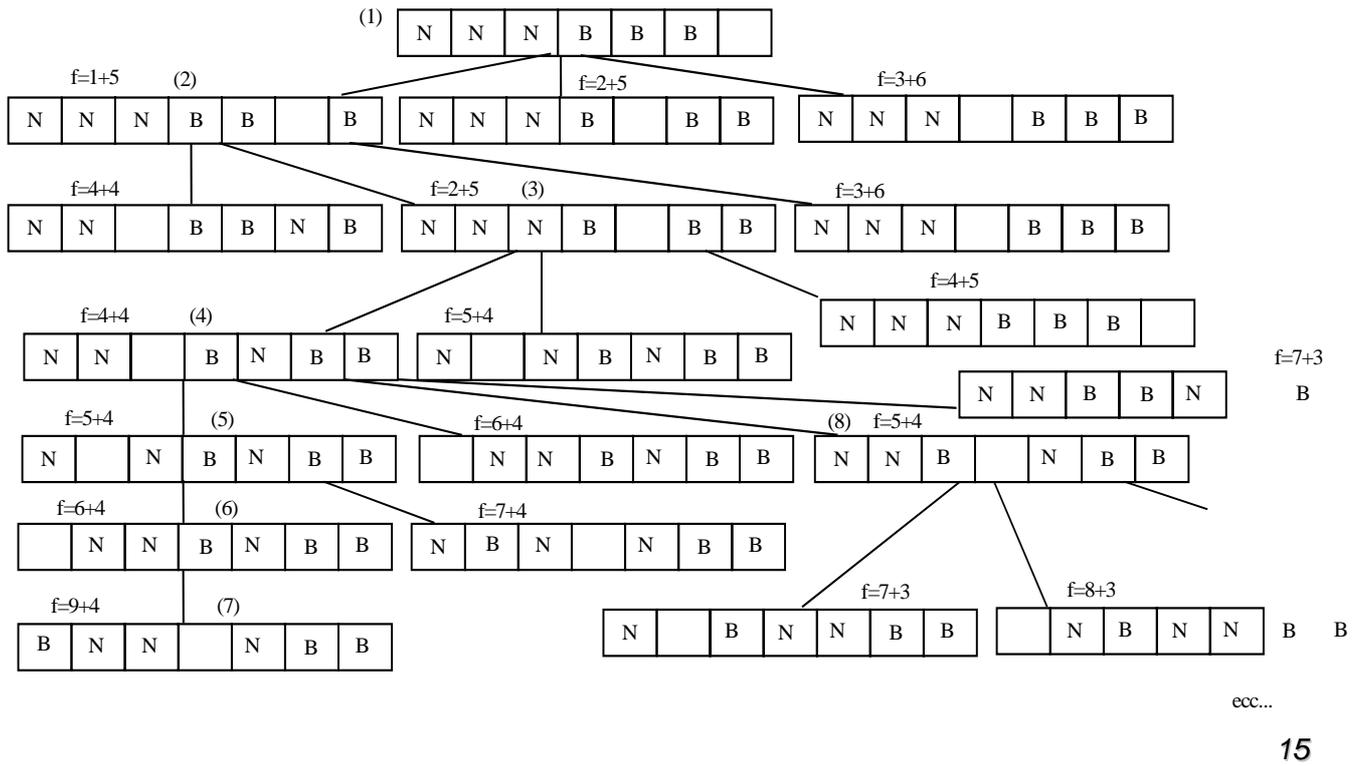
1	2	3	4	5	6	7
B	B	B		N	N	N

- $h=0$
- $h'=3$
- il costo dell'euristica è 3 mentre il costo vero è 0 essendo già arrivati al goal.

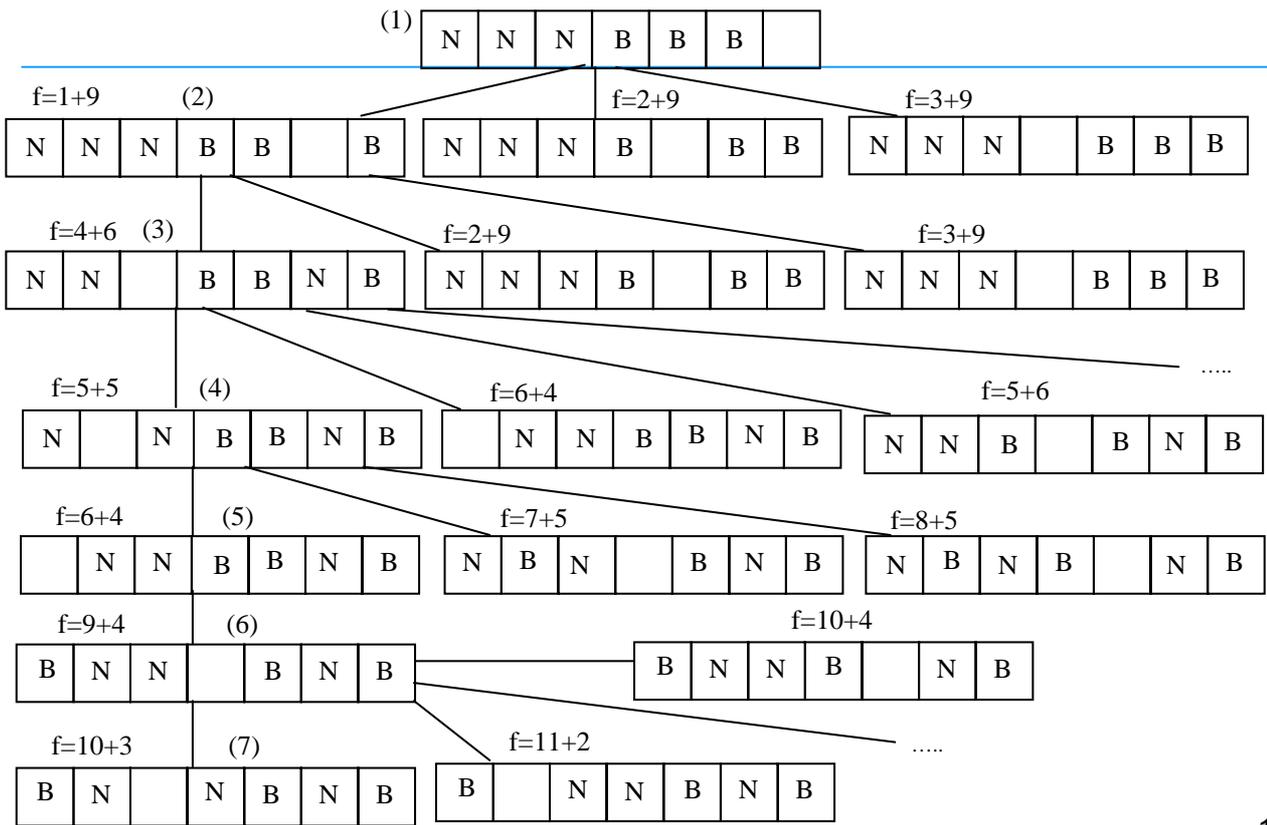
Esercizio – strategie di ricerca

- Euristica ammissibile: min distanza delle tessere nere da qualunque posizione goal. Si supponga di avere 12 monete ripartite in tre distinte pile. La prima pila è formata da 4 monete, la seconda da 7 e la terza da 1. Lo spostamento delle monete deve avvenire secondo la seguente regola: ogni volta che si spostano delle monete da una pila A ad un'altra B, le monete in B devono raddoppiare di numero.

Soluzione 1 – strategie di ricerca

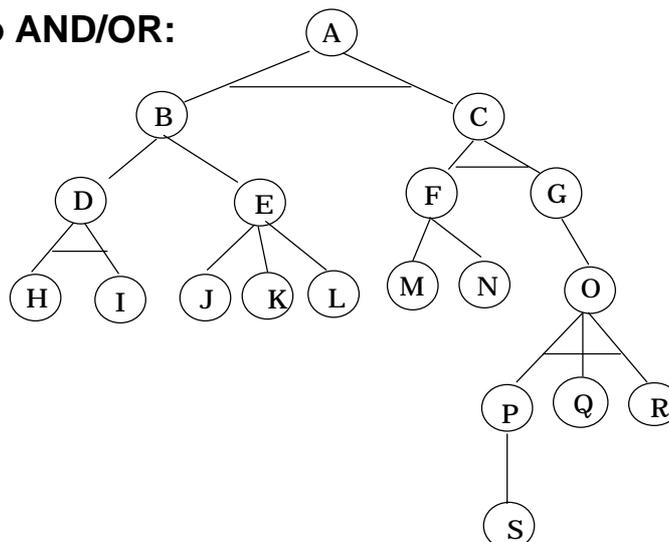


Soluzione 2 – strategie di ricerca



Esercizio – strategie di ricerca

- Dato il seguente albero AND/OR:

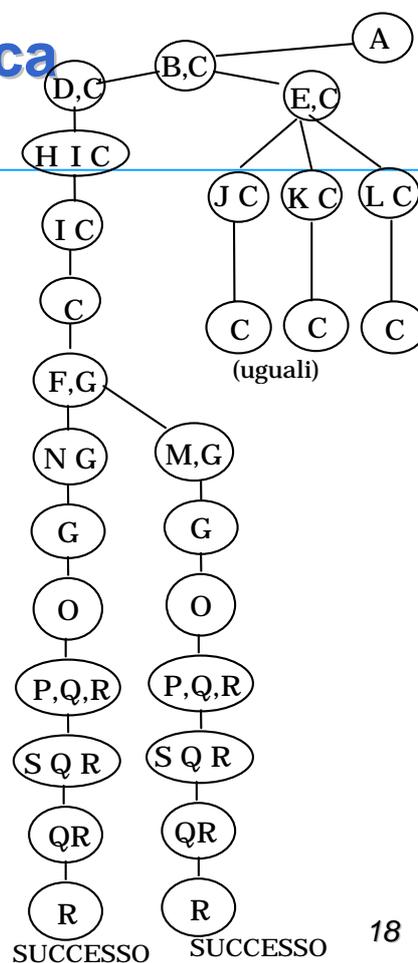


- Si indichino quali nodi devono essere di successo affinché venga dimostrato A. Si trasformi inoltre l'albero AND/OR in un albero OR equivalente, supponendo che tutte le foglie siano di successo. Esiste un solo albero OR equivalente o ne possono esistere più di uno?

17

Soluzione – strategie di ricerca

- I nodi che devono essere di successo affinché si dimostri A sono H e I oppure J o K o L assieme a M o N con S Q ed R.
- Può esistere più di un albero OR, ma non cambiano i rami di successo.

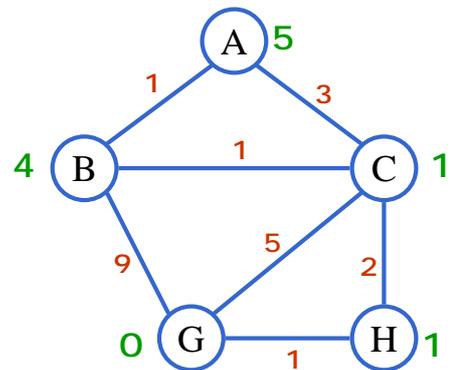


18

Esercizio – strategie di ricerca

- Si consideri il seguente grafo, in cui i numeri sugli archi sono i costi e i numeri vicino agli stati sono le stime euristiche h . Si noti che gli archi sono non orientati e quindi intesi come percorribili da entrambe le parti. Si consideri lo stato A come stato iniziale e G come lo stato Goal.

- Si simuli la ricerca A^* su questo grafo e si mostri l'albero generato supponendo di tenere traccia degli stati già espansi (ma non della strada ed il costo ad essi associato) per evitare loops.



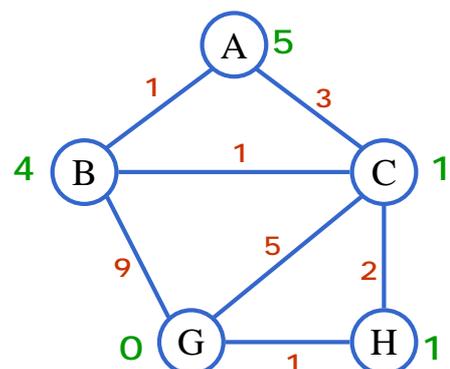
19

Esercizio – strategie di ricerca

- Si consideri il seguente grafo, in cui i numeri sugli archi sono i costi e i numeri vicino agli stati sono le stime euristiche h . Si noti che gli archi sono non orientati e quindi intesi come percorribili da entrambe le parti. Si consideri lo stato A come stato iniziale e G come lo stato Goal.

Si risponda alle seguenti domande:

1. L'euristica presentata è ammissibile?
2. L'euristica presentata è consistente?
3. L'algoritmo trova la strada ottima? (a minor costo?) Per quale motivo? Se no, quale specifico cambiamento di valori euristici dello stato sarebbe sufficiente per renderla consistente?

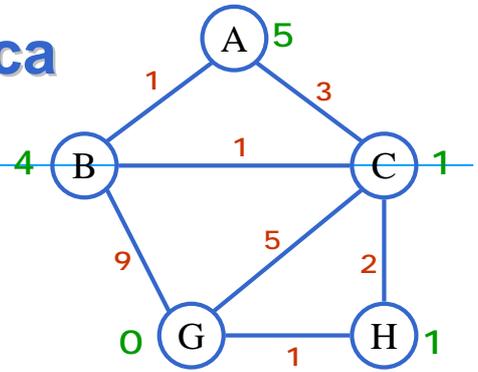


20

Esercizio – strategie di ricerca

Soluzione

Percorso attuale	Costo	Stima	Lista Nodi Espansi
A	0	5	



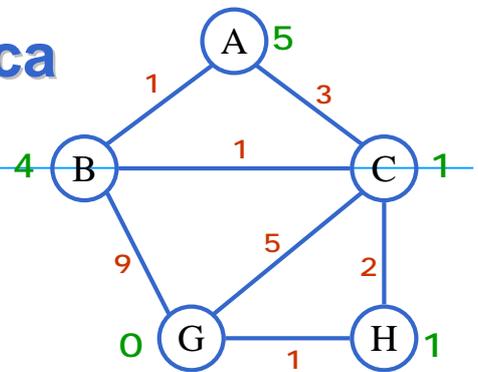
Scegliamo il nodo A

A
 $0 + 5 = 5$

Esercizio – strategie di ricerca

Soluzione

Percorso attuale	Costo	Stima	Lista Nodi Espansi
A	0	5	A



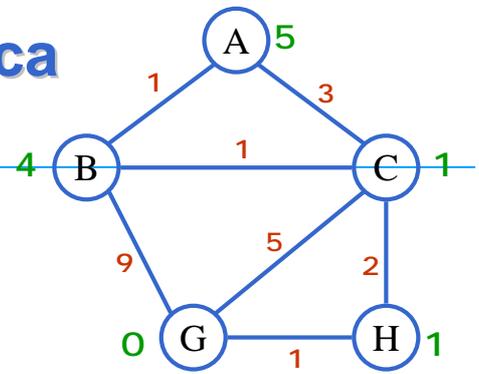
→ Scegliamo il nodo C

$5 = 1 + 4$ B C $3 + 1 = 4$

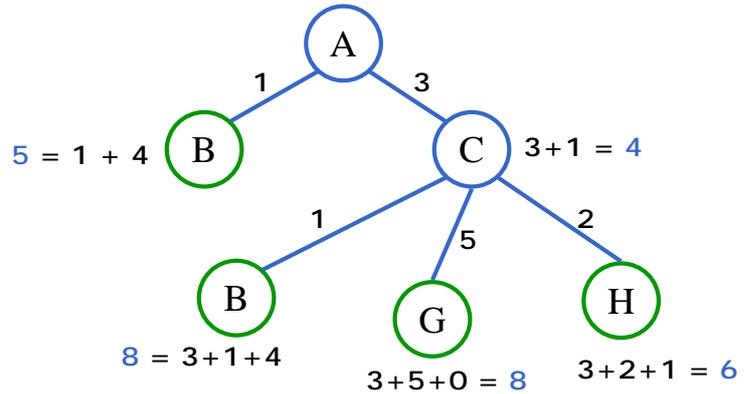
Esercizio – strategie di ricerca

Soluzione

Percorso attuale	Costo	Stima	Lista Nodi Espansi
A	0	5	A
CA	3	4	CA



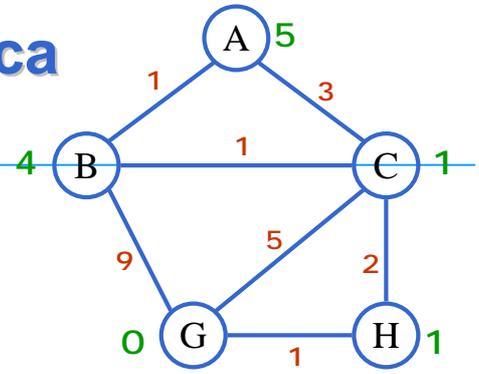
→ Scegliamo il nodo B
(percorso B A)



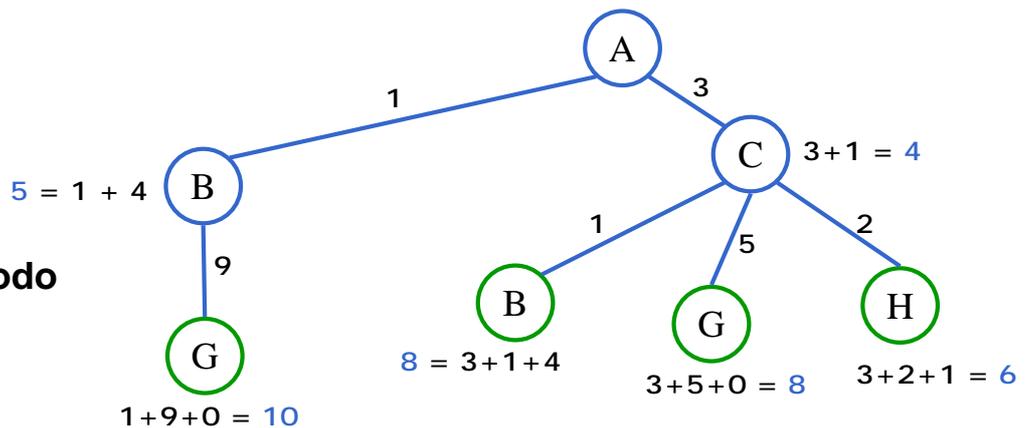
Esercizio – strategie di ricerca

Soluzione

Percorso attuale	Costo	Stima	Lista Nodi Espansi
A	0	5	A
CA	3	4	CA
BA	1	5	BCA

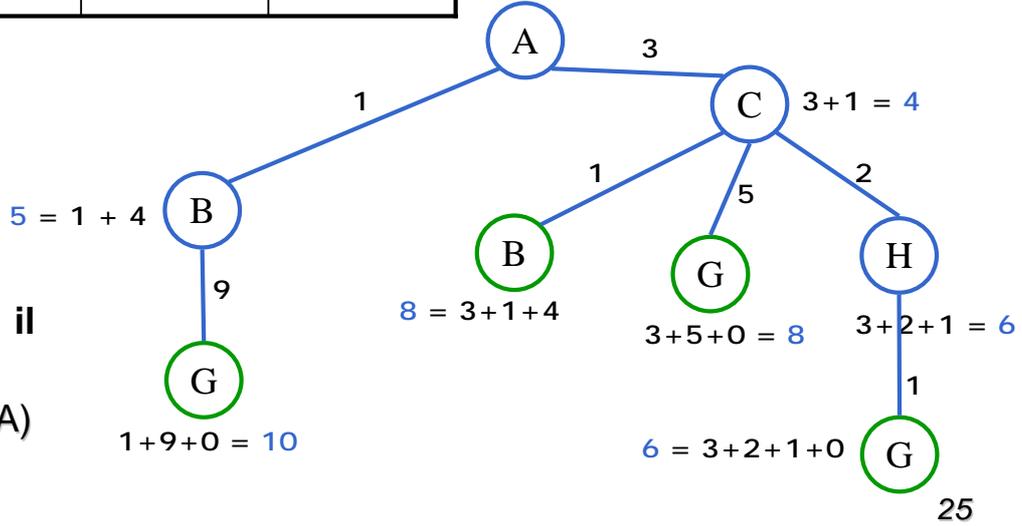
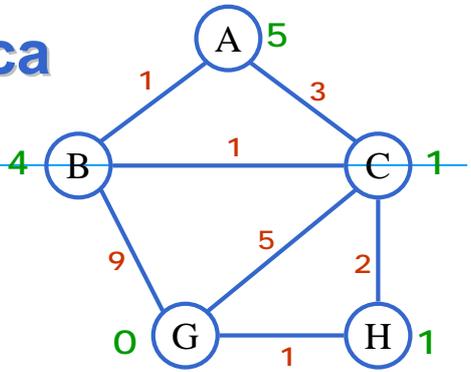


→ Scegliamo il nodo H
(percorso H C A)



Esercizio – strategie di ricerca Soluzione

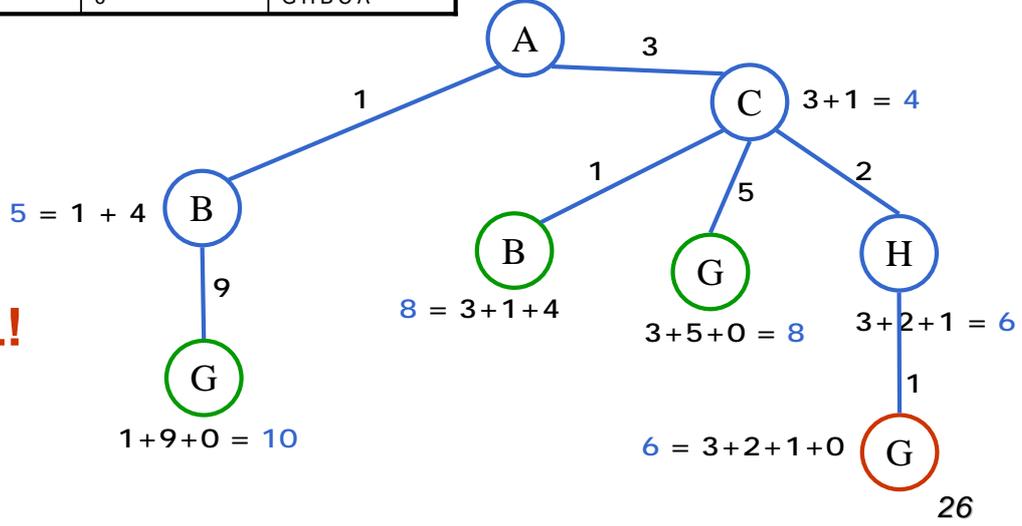
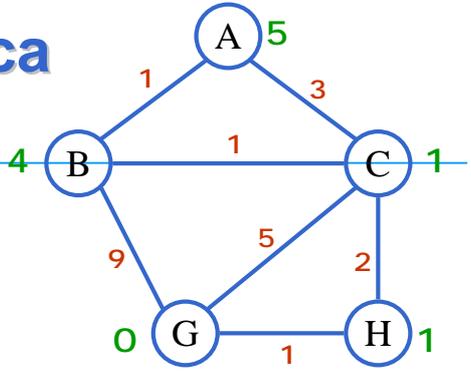
Percorso attuale	Costo	Stima	Lista Nodi Espansi
A	0	5	A
CA	3	4	CA
BA	1	5	BCA
HCA	5	6	HBCA



→ Scegliamo il nodo G (percorso G H C A)

Esercizio – strategie di ricerca Soluzione

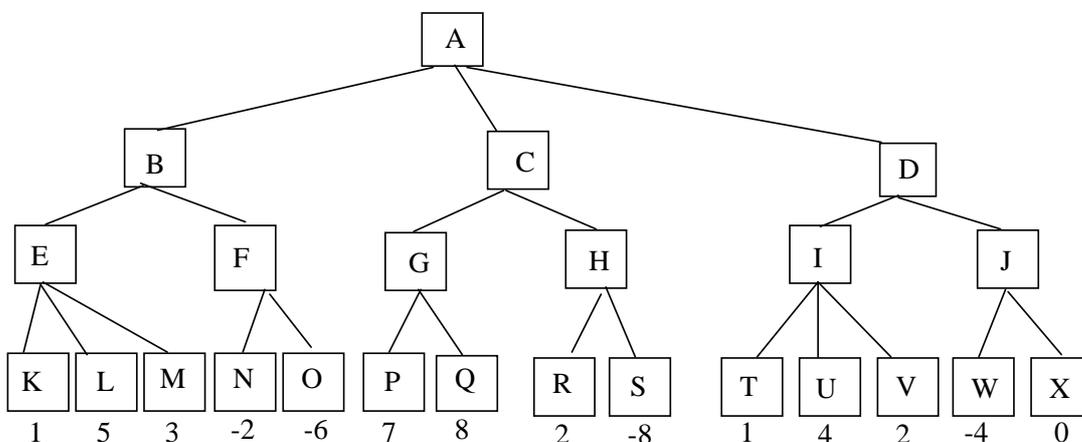
Percorso attuale	Costo	Stima	Lista Nodi Espansi
A	0	5	A
CA	3	4	CA
BA	1	5	BCA
HCA	5	6	HBCA
GHCA	6	6	GHBCA



G è il GOAL!

Esercizio – min-max e tagli alfa-beta

- Si consideri il seguente albero di gioco dove i punteggi sono tutti dal punto di vista del primo giocatore.

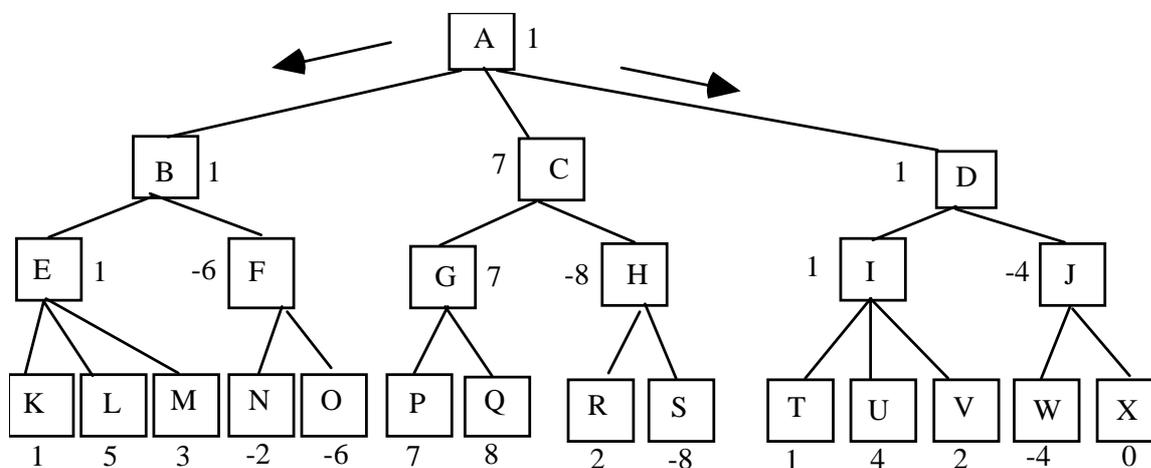


- Supponendo che il primo giocatore sia MIN, quale mossa dovrebbe scegliere? Si applichi l'algoritmo MIN-MAX. Successivamente si mostrino i tagli alfa-beta.

27

Soluzione – min-max e tagli alfa-beta

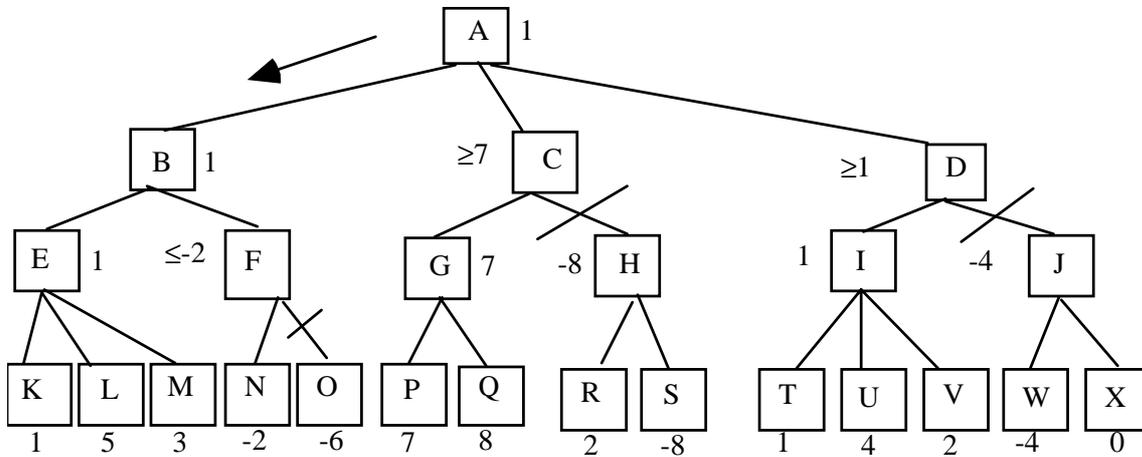
- Min-Max:



28

Soluzione – min-max e tagli alfa-beta

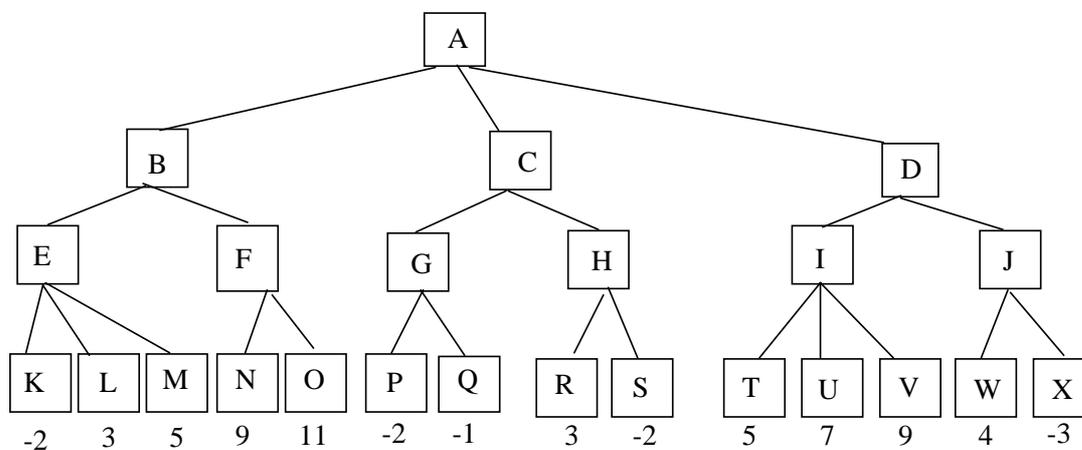
- Tagli alfa-beta:



29

Esercizio – min-max e tagli alfa-beta

- Si consideri il seguente albero di gioco dove i punteggi sono tutti dal punto di vista del primo giocatore.

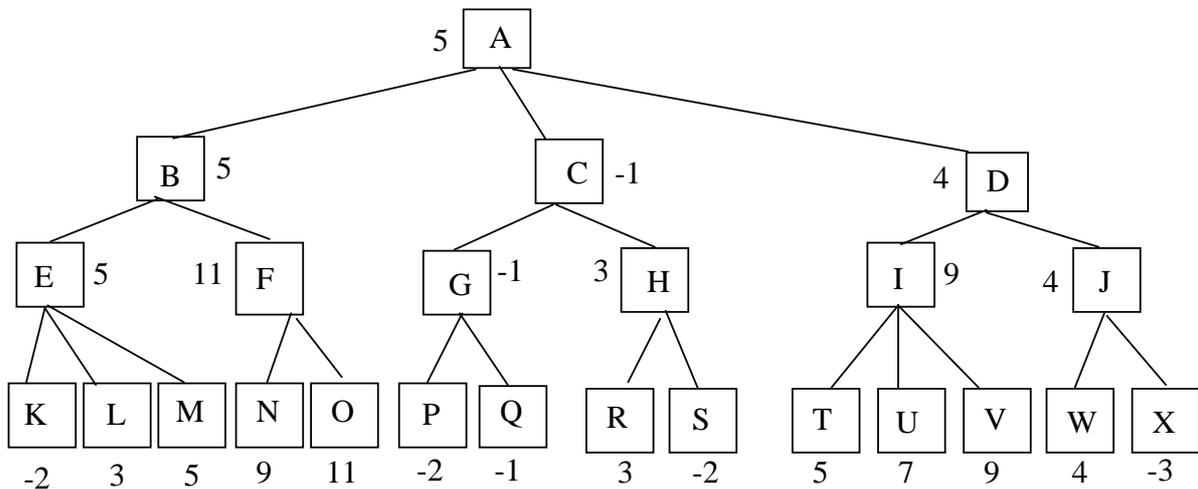


- Supponendo che il primo giocatore sia MAX, quale mossa dovrebbe scegliere? Si applichi l'algoritmo MIN-MAX. Successivamente si mostrino i tagli alfa-beta.

30

Soluzione – min-max e tagli alfa-beta

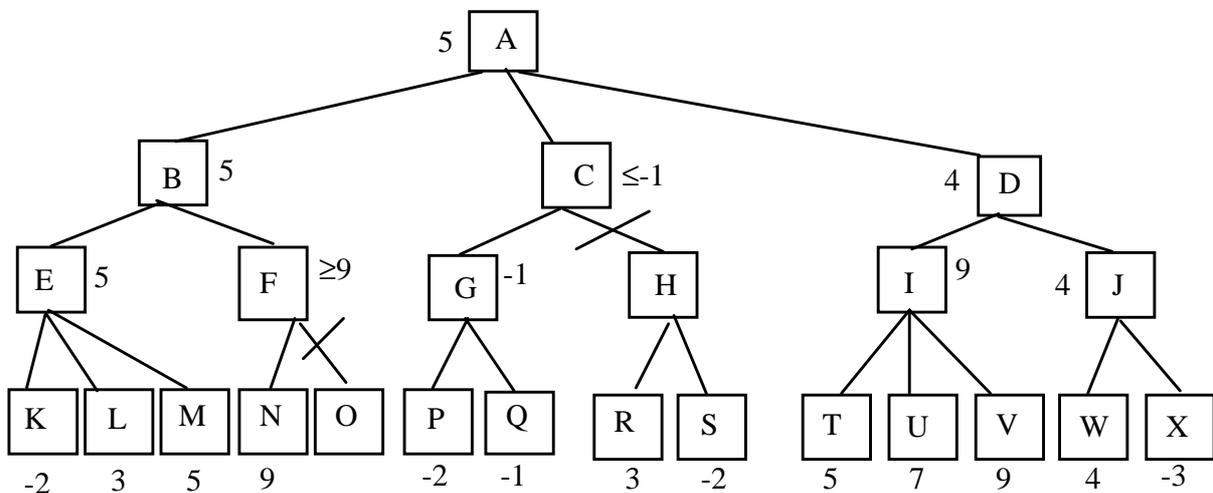
- Min-Max:



31

Soluzione – min-max e tagli alfa-beta

- Tagli alfa-beta:



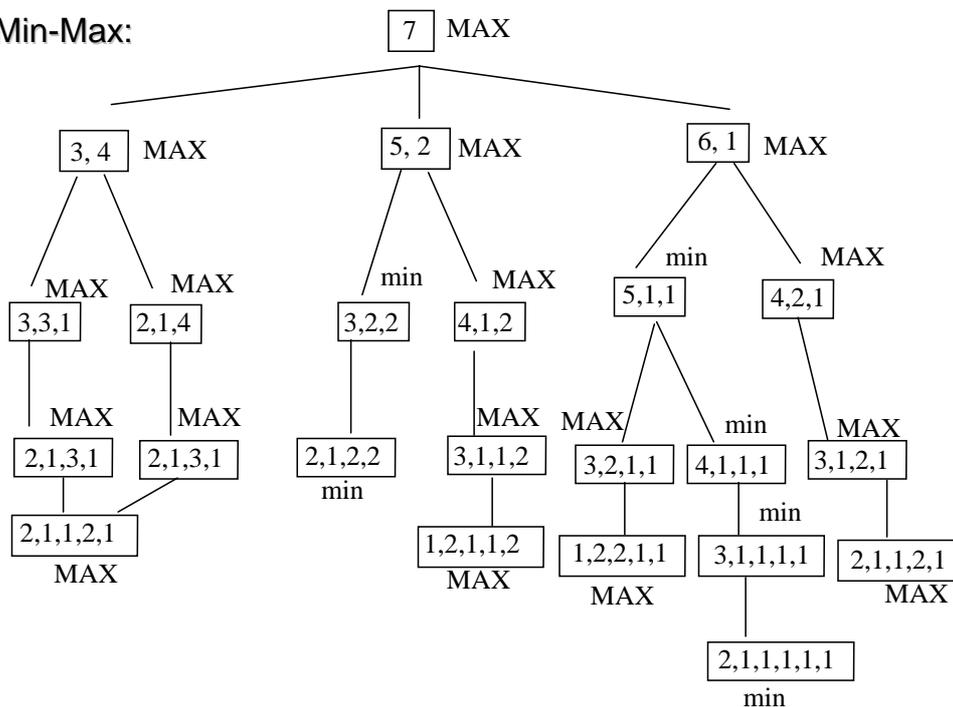
32

Esercizio – min-max e tagli alfa-beta

- Si consideri il seguente gioco:
- due giocatori hanno davanti una singola pila di oggetti (ad esempio una pila di monete).
- Il primo giocatore divide la pila in due pile separate che devono essere diverse. Poi continua il secondo lavorando su una a scelta delle due pile e così via.
- Il gioco termina solo quando in ogni pila ci sono una o due monete. A questo punto il primo giocatore che non può muovere ha perso.
- Si applichi la procedura MIN-MAX e si mostri lo spazio di ricerca nel caso in cui cominci MIN con una pila di 7 monete.
- Si commenti lo spazio risultante.
- Notazione: Ogni nodo dello spazio di ricerca rappresenta una configurazione: in particolare i numeri X,Y,Z indicano che si hanno tre pile corrispondenti ciascuna ad un numero, che indica il numero di monete contenute

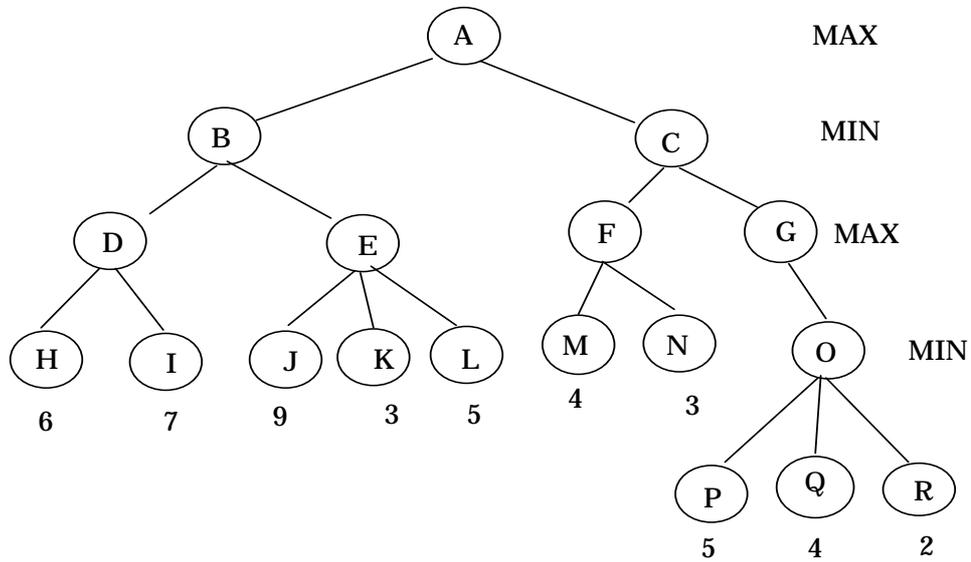
Soluzione – min-max e tagli alfa-beta

- Min-Max:



Esercizio – min-max e tagli alfa-beta

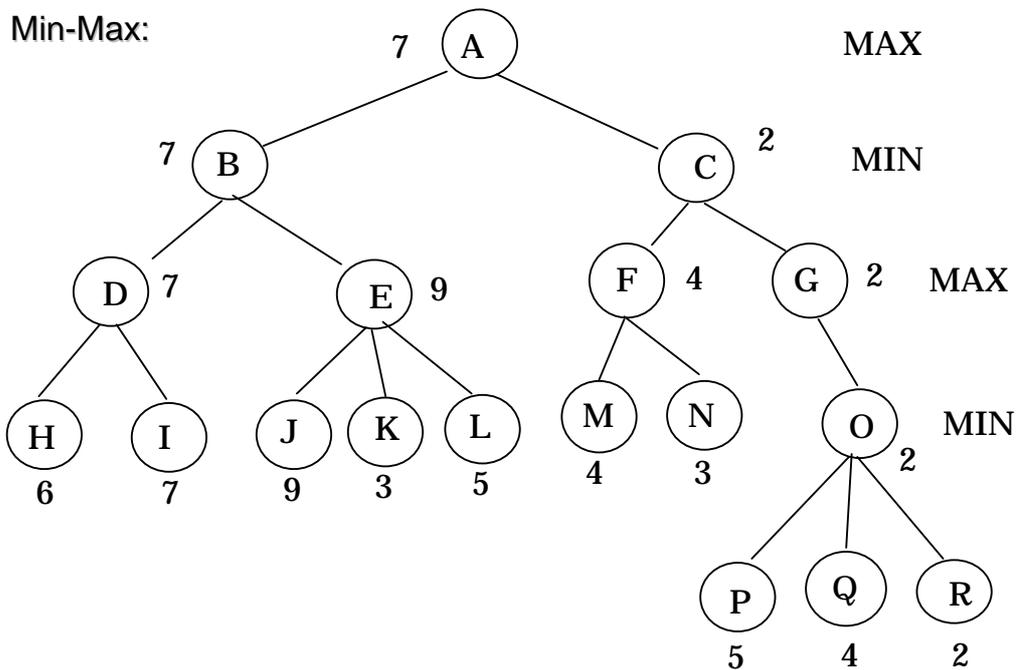
- Dato il seguente albero di ricerca per un gioco a due giocatori, si mostri quale mossa selezionerà MAX, secondo l'algoritmo MIN-MAX. Si mostri inoltre come può essere potato l'albero applicando i tagli alfa-beta.



35

Soluzione – min-max e tagli alfa-beta

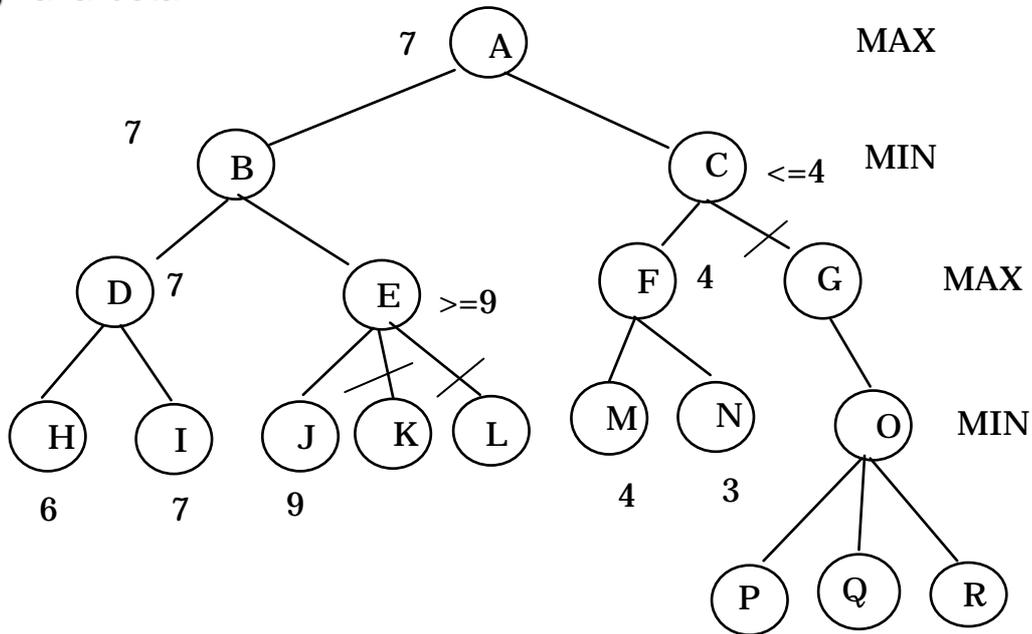
- Min-Max:



36

Soluzione – min-max e tagli alfa-beta

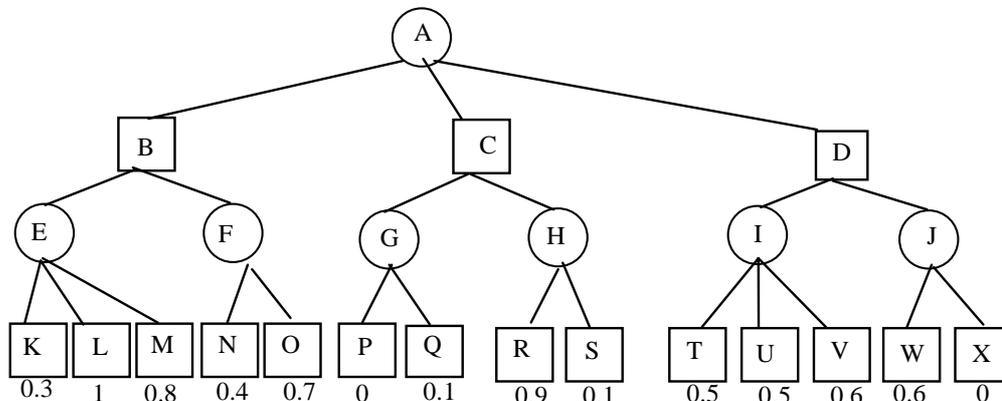
- Tagli alfa-beta:



37

Esercizio – min-max e tagli alfa-beta Compito del 5 Novembre 2003

- Si consideri il seguente albero di gioco dove i punteggi (compresi tra 0 e 1) sono tutti dal punto di vista del primo giocatore (0 indica la sconfitta di MAX e 1 la sua vittoria) :

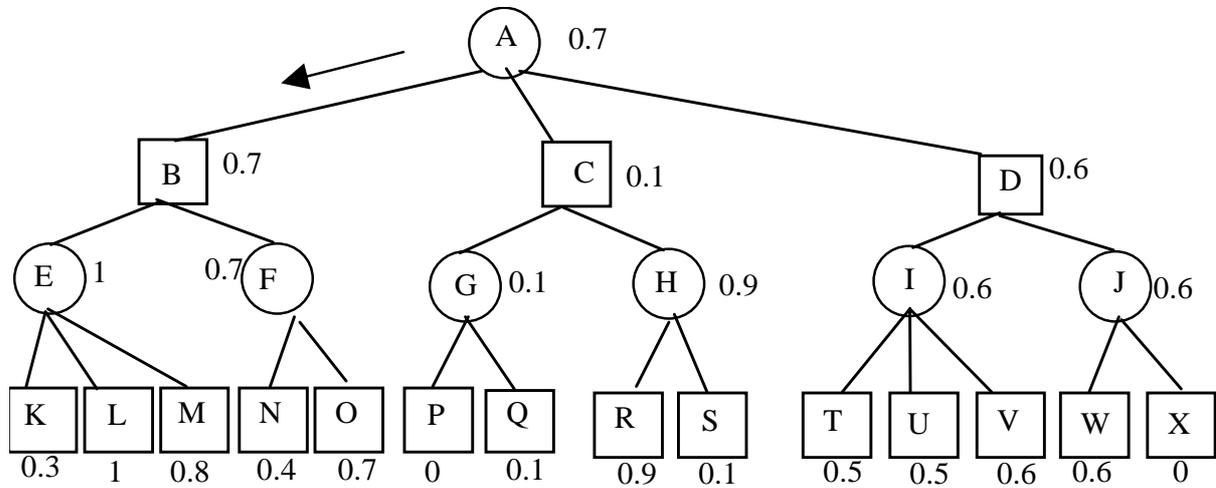


- Supponendo che il primo giocatore sia MAX, quale mossa dovrebbe scegliere? Si risolva l'esercizio tramite l'algoritmo MIN-MAX. Successivamente si mostrino i tagli alfa-beta.

38

Soluzione – min-max e tagli alfa-beta

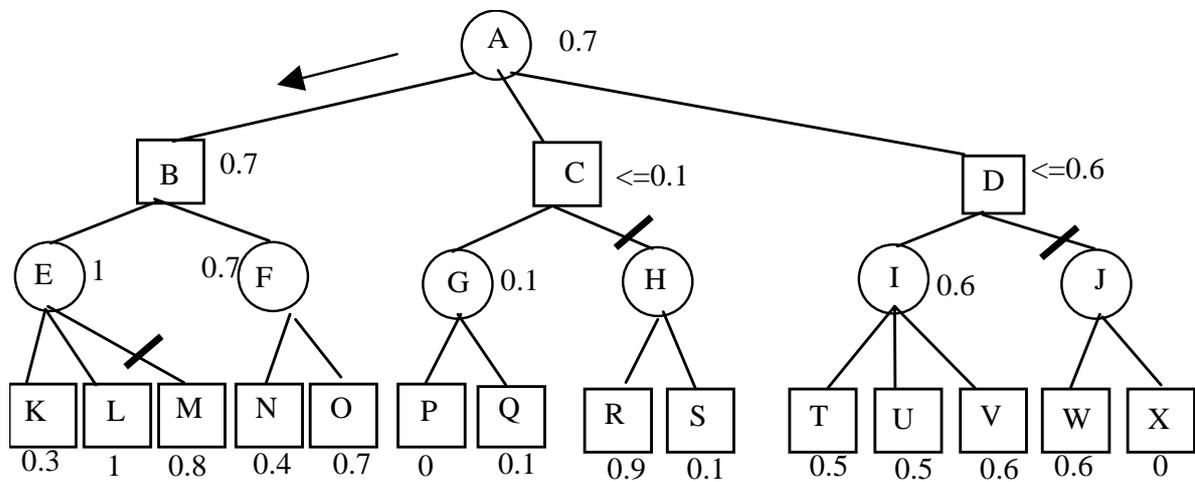
- Min-Max:



39

Soluzione – min-max e tagli alfa-beta

- Tagli alfa-beta:



40

Compito del 9 Settembre 2003 – Es. 4

Si scriva un programma Prolog `liststruct (Lin, Lout)` che data una lista di strutture con funtore principale `f` e arità 1 annidate a diversi livelli, restituisca in uscita una lista di interi che corrispondono alla profondità (grado di annidamento) di ciascun elemento. La profondità 0 (il grado di annidamento 0) e' determinato dalla costante `f` con arità 0.

Esempio:

```
?- liststruct([f(f(f)),f, f(f)], X).
```

darà come risposta:

```
Yes X=[2,0,1]
```

41

Compito del 9 Settembre 2003 – Es. 4 Soluzione

```
liststruct([], []).  
liststruct([X|Y], [N|Tail]):-  
    nesting(X,N), liststruct(Y,Tail).  
  
nesting(f,0).  
nesting(f(X),N):-  
    nesting(X,N1), N is N1 +1.
```

42

Compito del 5 Novembre 2003 – Es. 2

Si definisca un predicato `no_ripetuti(Xs, Ys, N)` che è vero se `Ys` è la lista degli elementi in `Xs` senza duplicazioni e `N` il numero di elementi ripetuti.

Esempio:

```
:- no_ripetuti([3,5,3,9,9,8,9], Ys, N).  
yes  
Ys=[5,3,8,9] N=3
```

43

Compito del 5 Novembre 2003 – Es. 2 Soluzione

```
no_ripetuti([], [], 0).  
no_ripetuti([X|Xs], Ys, N):-  
    member(X, Xs),  
    no_ripetuti(Xs, Ys, N1), N is N1 + 1.  
no_ripetuti([X|Xs], [X|Ys], N):-  
    not member(X, Xs),  
    no_ripetuti(Xs, Ys, N).  
  
member(X, [X|_]).  
member(X, [_|_]) :- member(X, _).
```

44

Compito del 9 Dicembre 2003 – Es. 3

Si scriva un programma Prolog `listindex(Lin1,Lin2,Lout)` che date due liste di elementi `Lin1` e `Lin2` contenenti numeri, produca una lista in uscita `Lout` contenente come elementi solo gli elementi della lista `Lin2` di posizione specificata da `Lin1`. Si noti che `Lin1` contiene sempre numeri ≥ 1 . Si scrivano esplicitamente tutti i predicati Prolog usati nella soluzione.

Esempio:

```
listindex([5,3,1], [9,20,11,5,19,21,0], X)
```

restituisce

```
X=[19, 11, 9]
```

45

Compito del 9 Dicembre 2003 – Es. 3 Soluzione

```
listindex([], L, []).  
listlindex([E1|Rest1], L2, [N3|Rest3]):-  
    estrai(E1, L2, N3),  
    listindex(Rest1, L2, Rest3).  
  
estrai(1, [A|B], A):- !.  
estrai(Nin, [A|B], Nout):-  
    N is Nin - 1, estrai(N, B, Nout).
```

46

Compito del 21 Giugno 2005 – Es. 1 (6 punti)

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*

47

Compito del 21 Giugno 2005 – Es. 2 (6 punti)

Si scriva un programma PROLOG che data in ingresso una lista di liste `ListIn`, dia in uscita una nuova lista di liste `ListOut` 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:

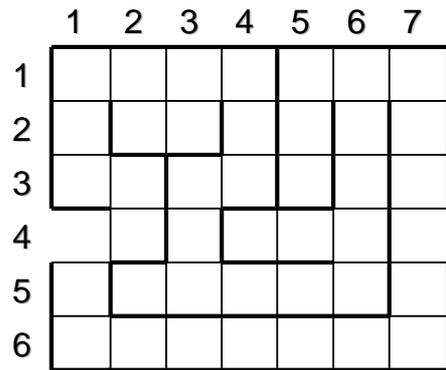
```
?-reduce([[1,5,2,4,9],[3,7,2],[9,3]],Z).
```

```
Z = [[1,5,2,4,9],[3,7],[9]]
```

48

Compito del 21 Giugno 2005 – Es. 3 (10 punti)

Dato il seguente labirinto rappresentato su una griglia 6x7:

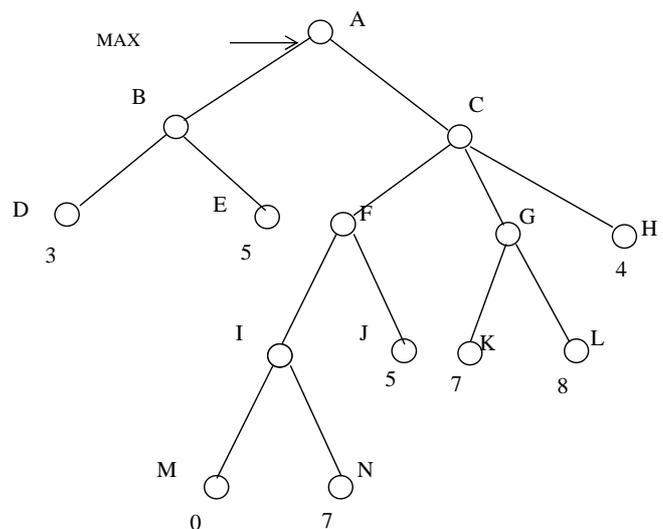


- formalizzare il problema di trovare un percorso dall'ingresso all'uscita come un problema di ricerca in uno spazio di stati;
- descrivere un algoritmo di tipo A* per i grafi per trovare la soluzione; *Suggerimento: si scelga come stima la Manhattan distance tra due punti $(x1,y1)$, $(x2,y2)$ che è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

49

Compito del 21 Giugno 2005 – Es. 4 (7 punti)

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:



- Evidenziare le valutazioni dei nodi mediante l'algoritmo MIN MAX e dire quale mossa farebbe MAX.
- 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?

50