Giochi

1. Sono giochi a due giocatori (di solito denominati min e max) in cui le mosse sono alternate e le funzioni di utilità complementari (vince e perde)
2. Sono giochi con conoscenza perfetta in cui i giocatori hanno la stessa informazione (non tipicamente i giochi di carte quali poker, bridge ecc)
3. Lo svolgersi del gioco si può interpretare come un albero in cui la radice è la posizione di partenza e le foglie le posizioni finali

MIN MAX

L’algoritmo min-max è progettato **per determinare la strategia ottimale per Max** e per **suggerirgli**, di conseguenza, **la prima mossa migliore da compiere**; per fare questo, ipotizza che Min faccia la scelta a lui più favorevole. Non è interessante la “strada”, ma solo la prossima mossa.

Le foglie (condizioni di fine gioco) sono etichettate con 1 e -1. Il giocatore **Min cerca di arrivare a -1** (minimizzatore), l'altro **max a +1** (massimizzatore).

Algoritmo Iterativo

Per valutare un nodo n:

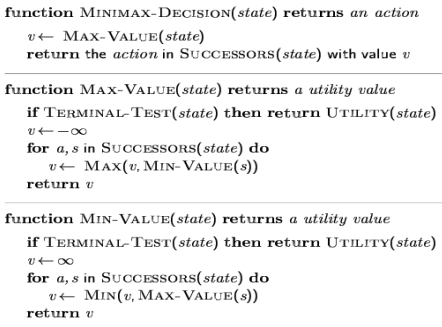
1. Espandi l'intero albero sotto la radice n
2. Valuta le foglie come vincenti per max o min
3. Seleziona un nodo n' senza etichetta i cui figli sono etichettati. Se non esiste alcun nodo senza etichetta, restituisci il valore assegnato ad n; altrimenti:
4. Se n' è un nodo in cui deve muovere min, assegna ad esso il valore minimo dei figli, se deve muovere max assegna il valore massimo dei figli. Torna a 3.

Patta: si assegna il valore 0.

Si possono assegnare dei valori ai nodi che poi vengono aggiornati quando si espandono i figli.

Complessità in tempo e in spazio (b, branching factor e d profondità dell’albero) O( bd )

Algoritmo Ricorsivo



TAGLI ALFA BETA

**Se il giocatore** (antagonista) **ha una scelta migliore M a livello del nodo genitore** od in un qualunque punto di scelta precedente, **N non sarà mai selezionato**. Se raggiungiamo questa conclusione possiamo eliminare N. → quindi eliminiamo N e tagliamo tutti i suoi discendenti non ancora considerati

1. α: valore della scelta migliore trovata sulla strada di MAX
2. β: valore della scelta migliore trovata sulla strada di MIN

L’algoritmo aggiorna ALFA e BETA, e taglia quando trova valori peggiori (rispetto ad ALFA, o rispetto a BETA), su strade alternative.

1. I valori (temporanei) nei MAX-nodes sono ALPHA-values
2. I valori (temporanei) nei MIN-nodes sono BETA-values

Principio di Alfa-Beta:

1. Se un Alpha-value è maggiore od uguale del Beta-value di un nodo discendente: stop alla generazione di figli del discendente
2. Se un Beta-value e’ minore od uguale all’Alpha-value di un nodo discendente: stop alla generazione dei figli del discendente

Algoritmo

Per valutare un nodo n in un albero di gioco:

1. Metti in L = [n] i nodi non ancora espansi
2. Sia x il primo nodo in L. Se x = n e c'è un valore assegnato a esso restituisci questo valore
3. Altrimenti se x ha un valore assegnato Vx, sia p il padre di x. Se a x non è assegnato un valore vai al passo 5
   1. Determiniamo se p ed i suoi figli possono essere eliminati dall'albero.

Se p è un nodo min, sia alfa il massimo di tutti i correnti valori assegnati ai fratelli di p (alfa del nodo max genitore) e dei nodi min che sono antenati di p (alfa dei nodi max antenati).

* 1. Se non ci sono questi valori alfa = - infinito.
  2. Se Vx <= alfa rimuovi p e tutti i suoi discendenti da L (dualmente con beta se p è un max).

Caso Duale

Determiniamo se p ed i suoi figli possono essere eliminati dall'albero.

Se p è un nodo max, sia beta il minimo di tutti i correnti valori assegnati ai

fratelli di p (beta del nodo min genitore) e dei nodi max che sono antenati di p (beta dei nodi min antenati).

Se non ci sono questi valori beta = + infinito.

Se Vx >= beta rimuovi p e tutti i suoi discendenti da L

Algoritmo ricorsivo

