Intelligenza artificiale applicata agli scacchi

Progetto per il corso “Intelligenza Artificiale” 2014/2015

Stefano Pisciella

stefano.pisciela@studio.unibo.it

Introduzione

Questa relazione andrà a descrivere l'implementazione di un intelligenza artificiale in una variante degli scacchi in cui ii giocatori hanno a disposizione solo pedoni e re e si vince promuovendo uno dei pedoni.
Il progetto è stato realizzato completamente da zero utilizzando il software framework Unity3D (http://unity3d.com).
Naturalmente l'analisi sarà ristretta solo alla descrizione dell'intelligenza sviluppata, non saranno approfonditi quindi aspetti come la grafica o la gestione delle mosse o delle pedine , etc…, in quanto esulano dall'argomento del progetto

Intelligenza artificiale

Funzione d'utilità

La funzione d'utilità è il nucleo dell'intelligenza artificiale, il suo scopo è quello di prendere come input lo stato attuale della scacchiera, in altre parole la posizione delle pedine sulla scacchiera, e restituire un valore intero che verrà poi utilizzato dall'algoritmo di ricerca per identificare la mossa migliore.
Per il lavoro realizzato, si è scelto di caratterizzare la funzione in due aspetti: *statico e dinamico.*
Per aspetto *statico* si intende un valore che viene dato in modo 'fisso' alle pedine e che non cambia per tutta la durata della partita. In pratica in tabella 1 e 2, rispettivamente pedoni e re, sono osservabili i valori assegnati alla pedina in base alla posizione. Il valore statico della funzione di utilità sarà quindi la somma dei valori delle posizioni occupate dalle pedine.
Quindi ad esempio il valore statico iniziale relativo ad un giocatore sarà:

pedoni = 110x6 + 80x2, re = -60, totale = 760

Tab1: Valutazioni dei pedoni in base alla posizione

8	*100*	*100*	*100*	*100*	*100*	*100*	*100*	*100*
7	*170*	*170*	*170*	*170*	*170*	*170*	*170*	*170*
6	*150*	*150*	*150*	*150*	*150*	*150*	*150*	*150*
5	*130*	*130*	*130*	*130*	*130*	*130*	*130*	*130*
4	*125*	*130*	*130*	*130*	*130*	*130*	*130*	*125*
3	*110*	*110*	*110*	*110*	*110*	*110*	*110*	*110*
2	*110*	*110*	*110*	*80*	*80*	*110*	*110*	*110*
1	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**

Tab2: Valutazioni del re in base alla posizione

8	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
7	*30*	*30*	*30*	*30*	*30*	*30*	*30*	*30*
6	*40*	*40*	*40*	*40*	*40*	*40*	*40*	*40*
5	*40*	*40*	*60*	*60*	*60*	*60*	*40*	*40*
4	*40*	*40*	*60*	*60*	*60*	*60*	*40*	*40*
3	*40*	*40*	*40*	*45*	*45*	*40*	*40*	*40*
2	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
1	*-60*	*-60*	*-60*	*-60*	*-60*	*-60*	*-60*	*-60*
	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**

Se analizzate bene, queste tabelle definiscono uno dei comportamenti standard nei finali "Pedoni e Re vs Pedoni e Re", ovvero quello di cercare, se possibile, di portare subito il re al centro muovendo prima di tutti uno dei pedoni centrali per fargli strada. Dopodichè la partita viene 'lasciata' ai pedoni che, sono spinti ad andare avanti sotto la 'protezione' del re.
Dal momento che può sembrare strano, va specificato che la riga 8 della tabella dei pedoni, che segna la vittoria nel gioco, ha un valore basso poichè, se un pedone si trova in una di quelle posizione, il gioco rileva uno stato terminale che assegna in automatico un valore estramemente alto.

Per aspetto *dinamico* si intende un valore che viene dato alle pedine solo in base al rispetto di alcuni pattern generali, come potrebbe essere una particolare 'forma geometrica' creata dalle pedine in base alla loro posizione sulla scacchiera. In questo progetto sono stati introdotti due concetti dinamici:

- *Allineamento dei pedoni,* I pedoni acquistano valore se hanno nelle caselle adiacenti, ad eccezione di quella avanti e quella dietro, altri pedoni alleati. Viene dato maggior valore ai pedoni sulla stessa linea rispetto a quelli adiacenti in diagonale.
- *Pedone passato,* Per pedone passato si intende un pedone che davanti a sé, nella sua colonna e in quelle adiacenti, non ha pedine avversari. A questo tipo di pedone viene dato un valore crescente, in modo esponenziale, in base alla vicinanza alla vittoria. Questo per far si che una volta creato un pedone passato, si preferisca mandarlo avanti piuttosto che creare un altro pedone passato.

Naturalmente le considerazioni sopra, sia dinamiche che statiche, vengono effettuate per entrambi i giocatori e il valore finale della funzione d'utilità sarà la differenza tra il valore delle

pedine del giocatore guidato dall'intelligenza artificiale e quelle dell'avversario. In questo modo l'avversario vorrà minimizzare l'utilità mentre l'intelligenza cercherà di massimizzarla, aspetto che ci porta verso l'algoritmo di ricerca della soluzione migliore.

Ricerca

Come algoritmo di ricerca si è scelto di utilizzare l' Alpha-beta iterativo con un limite di tempo *t* fissato in base alla difficoltà del gioco. Per spiegarlo meglio, di seguito lo pseudocodice dell'algoritmo (escludendo l'implementazione dell'alpha-beta).

```
PROCEDURE ALPHA-BETA-ITERATIVE-LIMITED-TIME (t)

begin_time = get_currenttime()
depth = 1
current_solution

WHILE true DO

      solution = alpha-beta(depth, begin_time, t)

      IF get_currenttime() <= begin_time + t
            current_solution = solution
            depth = depth + 1
      ELSE
            break;

RETURN current_solution
```

Nell'algoritmo sopra si utilizza una funzione *get_currenttime* che restituisce il tempo corrente e si utilizza una variante dell'alpha-beta che si ferma alla profondità *depth* e che termina la computazione, restituendo un risultato non valido, se è passato più di *t* tempo da *begin_time*.

Si è scelto di utilizzare un approccio iterativo rispetto ad uno con profondità fissata per due motivi principali:

- Dai svariati test, il livello raggiunto dall'iterativo è almeno uguale al massimo livello raggiungibile fissando la profondità nell'intera partita. Questo perchè sappiamo che una ricerca depth-first (come è l'alpha-beta) su un albero uniforme di profondità *d* e fattore di branching *b* impiega tempo $O(b^d)$ che è lo stesso tempo che impiega una ricerca depth-first iterativa con livello massimo *d.* Questo però ci garantisce una scalabilità migliore nel caso di un miglioramento delle prestazioni del calcolatore.
- Si è notato che il giocatore medio man mano che il gioco va avanti, e che di conseguenza le pedine diminuiscono, riesce a guardare più avanti di quanto il nostro algoritmo con livello fissato riesca a fare. Grazie al fattore iterativo però, l'intelligenza artificiale riesce a comportarsi proprio come il 'giocatore medio' ovvero a guardare più avanti impiegando lo stesso tempo di computazione.

In pratica il fattore principale che fa preferire l'utilizzo di una ricerca iterativa è che quando le pedine sono tante e ci sono tante mosse possibili la profondità massima raggiungibile è limitata (per tempo e spazio). Quindi si dovrebbe fissare una profondità massima raggiungibile nello stato iniziale (inizio della partita), con un tempo ragionevole e nello spazio disponibile, e tenere quello per tutta la partita. Ma questo implica una forte limitazione nel

momento in cui le pedine e le mosse disponibili sono molto meno e si potrebbe guardare avanti per un numero di mosse maggiore.

Conclusione

In conclusione il prototipo creato si comporta egregiamente con giocatori di scacchi di medio/basso livello, vincendo la maggior parte delle volte (nei test effettuati). La percentuale di vittorie cade drasticamente nel caso di giocatori di medio/alto livello, probabilmente anche a causa della facilità della variante degli scacchi implementata che permette loro di guardare e ragionare molto più avanti dell'intelligenza.
Ci sono due aspetti su cui si può ancora lavorare per migliorare il progetto:

- L'ottimizzazione della ricerca, per esempio cercando di dare un ordine agli stati da valutare garantendo una maggiore efficacia della potatura alpha-beta e di conseguenza permettere all'intelligenza di guardare più avanti.
- Il rafforzamento dell'aspetto dinamico della funzione d'utilità. La ricerca a tempo limitato ci permette di creare funzioni di utilità molto più complesse (a discapito naturalmente della profondità raggiunta).