

Algoritmo Minimax, varianti ed euristiche: applicazione al gioco Dots and Boxes

Riccardo Angius, matricola 1074854

Università degli Studi di Padova
Corso di Laurea Magistrale in Informatica
Insegnamento di Intelligenza Artificiale

A.A. 2013/2014

Regole del gioco

- ▶ Due giocatori.
- ▶ Una linea tra due punti adiacenti è tracciata ad ogni turno.
- ▶ Il gioco finisce quando tutti i punti sono collegati.
- ▶ Il vincitore è colui che ha tracciato più linee per ultimo nella composizione di un quadrato.



► Ambiente: Python + WxPython + NetworkX

1. Applicazione per sfida tra utenti
2. Agente minimax naive
3. Pruning, cutoff e valutazione
4. Tabella delle trasposizioni
5. Ordinamento dei successori
6. Ricerca con backtracking
7. Ricerca ad approfondimento iterativo
8. Tempo reale
9. Ricerca quiescenza

Funzioni di valutazione euristiche

$$Eval_1(P) = S_{4A} - S_{4B}$$

$$Eval_4(P) = 2(S_{4A} - S_{4B}) + s(P)(0.75S_3 - 0.5S_2)$$

$$s(P) = \begin{cases} +1 & P = A \\ -1 & P = B \end{cases}$$

- ▶ S_N : gruppi di punti con N linee tracciate su 4
- ▶ S_{4P} : punteggio del giocatore P

Funzioni di valutazione euristiche

Sperimentali

$$Eval_5(P) = 2(S_{4A} - S_{4B}) + s(P)(S_2)$$

$$Eval_6(P) = 2(S_{4A} - S_{4B}) + s(P)(S_3)$$

$$Eval_7(P) = 2(S_{4A} - S_{4B}) + s(P)(S_4)$$

- ▶ S_N : gruppi di punti con N linee tracciate su 4
- ▶ S_{4P} : punteggio del giocatore P

Risultati sperimentali

100 partite, 1 secondo a ogni turno per decidere

Esperimento	1	2	X
C_1 vs $C_1 T_1$	30	40	30
C_1 vs $C_1 T_2$	27	32	41
$C_1 T_1$ vs $C_1 T_2$	40	27	33
C_1 vs $C_1 Q$	26	45	29
C_1 vs C_4	38	24	38
$C_1 Q$ vs C_5 e	33	36	31
$C_1 Q$ vs C_6	28	33	39
$C_1 Q$ vs C_7	29	32	39
C_5 vs C_6	36	29	35
C_5 vs C_7	34	28	38
C_6 vs C_7	31	30	39

- ▶ C_K : agente usa $Eval_K$
- ▶ Suffisso T_j : agente usa ordinamento successori
- ▶ Suffisso Q : agente usa ricerca quiescenza

Risultati sperimentali

100 partite, 1 secondo a ogni turno per decidere

- ▶ C_1Q meglio di $C_1 \dots$
- ▶ \dots ma non tanto meglio di C_5 , C_6 e C_7 !
- ▶ C_4 fuori strada!

- ▶ Sperimentazione con turni più lunghi, tavoliere più grande
- ▶ Nuove euristiche sulla base della long chain rule e simili
- ▶ Modifica incrementale stato
- ▶ Certificati isomorfismo grafo rappresentate stato