

Exemplo de poda α - β e função heurística para damas

Emmanuel Podestá Jr., Gustavo Zambonin*

Inteligência Artificial (UFSC – INE5430)

- 1) Em anexo ao final do documento, a árvore resolvida a partir do algoritmo de poda. Para os nodos folha, α e β não foram computados para que a imagem não ficasse confusa; locais marcados com um asterisco (*) são passíveis de poda caso existissem outros filhos.
- 2) Considerando um jogo de damas usual e observando um modelo de grafo convencional, que mapeia estados do tabuleiro para nodos e jogadas válidas conectando-os como arestas, é possível observar que tal árvore resultante impossibilitará o uso de uma função utilidade apenas, por conta do seu vasto número de possibilidades para jogadas.

Analisando as regras do jogo, alguns tópicos foram escolhidos de modo a diferenciar um tabuleiro que representa uma boa jogada. Estes podem ser utilizados para ambos os tipos de funções.

- É necessário normalizar o valor do tabuleiro em relação ao jogador humano, então pode-se decrementar da pontuação deste o valor obtido pela análise das peças do jogador computacional.
- O número de “damas” do jogador deve influenciar mais no valor final do tabuleiro do que peças normais, visto que é mais difícil obtê-las e sua capacidade de movimentação e ataque são melhoradas.
- O número de peças no tabuleiro é um fator a ser considerado, visto que é muito mais difícil vencer um jogo se suas peças estão cercadas por outras de cor diferente.
- As possibilidades de ataque de peças normais e “damas” devem ser consideradas, de modo que o algoritmo deve verificar se existem movimentos que possam remover uma ou mais peças do adversário; porém, tais verificações serão mais custosas em virtude do incremento da procura em profundidade para considerar ataques duplos ou triplos.

Note que tais análises são extremamente ingênuas; para produzir um jogador computacional hábil, recursos como a pré-computação de valores para estados de tabuleiros específicos, na forma de bancos de dados; grandes listagens de movimentos de abertura, para maximizar a chance de boas jogadas desde o começo; e otimizações como tabelas de transposição¹ podem ser inseridos no algoritmo para que seu conhecimento seja aumentado. Todos estes exemplos foram aplicados no jogador *Chinook* [1].

Referências

- [1] Jonathan Schaeffer and Robert Lake. Solving the game of checkers. In *Games of No Chance*, volume 29, pages 119–136. Cambridge University Press, 1996.

*{emmanuel.podesta,gustavo.zambonin}@grad.ufsc.br

¹tabelas que guardam estados de tabuleiro que são iguais e alcançáveis por um grande número de jogadas diferentes; ou seja, caso uma jogada já tenha sido descartada e um tabuleiro similar aparece novamente, uma consulta nesta tabela pode acelerar a procura de possibilidades no algoritmo, descartando este tabuleiro.

