RELATÓRIO SOBRE O ALGORITMO MINIMAX

CIC 260 - INTELIGÊNCIA ARTIFICIAL

Nome:	Willian Saymon da Silva
Matrícula:	33962
Linguagem:	Python 3

O algoritmo minimax(ou minmax) é uma técnica para minimizar as chances máximas de derrota. Para isso ele desce até as folhas da árvore de jogadas(ou Game Tree) e verifica qual o caminho mais seguro para a vitória certeira.

A idéia geral do algoritmo é dividir a árvore de jogadas em níveis que corresponderão aos turnos dos jogadores. Os níveis são então classificados alternadamente em MAX e MIN. A partir das folhas a árvore começa a ser preenchida de maneira com que os níveis MIN armazenem o valor mínimo de seus filhos e os níveis MAX os valores máximos. Sendo que MAX corresponde ao turno da AI e MIN corresponde ao jogador que a enfrenta. O sentido de pegar o maior valor dos filhos nas camadas MAX é para maximizar as chances de vitória e o menor nos níveis MIN é simular a melhor jogada do oponente.

Como já comentado, é necessário descer até as folhas da árvore para poder obter a melhor jogada, de certa forma esse é um problema ,pois muita memória e tempo são gastos neste processo, devido a isso o minimax não pode ser utilizados para jogos ou situações com opções de jogadas muito vastas. Uma adaptação para solucionar esse procedimento é adotar uma profundidade limite de análise, assim não é necessário verificar toda a árvore, mas vale notar que este método pode levar para caminhos que no momento parecem bons, mas que no futuro são ruins. Para essa análise em determinada profundidade é necessária uma função de avaliação que classifique os estados e gere valores computáveis para os mesmos.

Para este trabalho foi desenvolvido um jogo da velha em python 3. O jogo utiliza o algoritmo minimax que analisa toda a árvore de jogadas que é gerada de maneira recursiva toda vez que é necessário determinar a próxima jogada a ser feita. É possível realizar algumas melhorias visando diminuir o tamanho do algoritmo, mas esta versão foi mantida por ser a primeira a ser desenvolvida para o trabalho. Inúmeros testes com diversos jogadores foram realizados, o algoritmo provou gloriosamente seu funcionamento não perdendo nenhuma partida sequer. Segue abaixo o mesmo em pseudocódigo:

```
1.  minimax(estado, turno, raiz)
2.      Se jogo acabou:
3.          retorna valor do resultado # 1 vitória / -1 derrota / 0 empate
4.      Se é o turno da AI:
5.          prox_turno ← jogador
6.          melhor_jogada ← [Null, -∞]
7.      Se não:
8.          prox_turno ← AI
9.          melhor_jogada ← [Null, ∞]
10.     Para cada opção_jogada em estado:
11.         prox_estado ← estado.jogar(opção_jogada)
12.         jogada ← [opção_jogada, minimax(prox_estado, prox_turno, falso)]
13.         Se é o turno da IA:
14.             Se melhor_jogada[1] < jogada[1]:
15.                 melhor_jogada = jogada
16.         Se não:
17.             Se melhor_jogada[1] > jogada[1]:
18.                 melhor_jogada = jogada
19.     Se raiz:
20.         retorna melhor_jogada[0]
21.     Se não:
22.         retorna melhor_jogada[1]
```

O algoritmo realiza uma busca em profundidade de maneira recursiva onde a condição de parada é determinada verificando se alguém venceu o jogo ou deu empate. Caso o estado atual seja um estado final ele retornará o valor de -1 caso seja uma derrota para a IA, 0 caso seja um empate e 1 caso seja um vitória. Caso não seja um estado final é necessário estabelecer a variável de melhor jogada com base no turno, se o turno for da IA, o estado corresponde a um nível MAX, se for do oponente, o estado corresponde a um nível MIN. Tendo verificado se está em um nível MAX ou MIN o algoritmo parte para etapa de análise dos nós filhos, para isso ele executa ele mesmo, o minimax, nos estados filhos do estado atual. Após obter o resultado de uma jogada é realizada uma comparação com a melhor jogada atual, e caso esta nova jogada analisada seja melhor ela é armazenada. Por fim se todas as funções foram finalizadas e removidas da pilha de execução a primeira chamada do minimax retorna a posição da jogada.

BIBLIOGRAFIA:

HEINEMAN, George T.; POLLICE, Gary; SELKOW, Stanley. **Algorithms in a nutshell: A practical guide**. " O'Reilly Media, Inc.", 2016.