UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
DEPARTAMENTO DE ENGENHARIA DA COMPUTAÇÃO E AUTOMAÇÃO

COMPONENTE: DCA0413 - CONTROLE INTELIGENTE

Marco Antonio Moreira Carujo

Professor: Fabio Meneghetti Ugulino De Araujo

Natal / RN

Abril de 2018

Sumário

Sumário.....2

1. Introdução..... 3

2. O Jogo da Velha..... 4

3. Jogando..... 6

Conclusão.....9

Referências.....9

1

Introdução

O projeto é a realiazação em um Jogo da Velha Eletrônico, feito em liguagem de programação Python, o qual utiliza a estrátegia do Minmax.

O Minmax por sua vez é uma teoria (método) de tomada de decisão, sendo um método para minimizar a possível perda máxima. Pode ser considerado como a maximização do ganho mínimo (maximin). Na nossa aplicação teremos dois jogadores (um sendo o computador). O usuário do jogo pode começar jogando ou não, e assim o computador tomará decisões e fará a sua jogada.

Além da utlização do método de tomada de decisão utilizamos algumas regras específicas para auxiliar uma decisão mais eficiente.

A estrátegia do ponto de vista de programação, foi com a utilização de Programação Orientada a Objetos, construindo 3 classes para que juntas fossem capazes de fazer as funcionalidades do jogo, além de sempre utilizar a recursão para efetuar sua lógica de programação, como será apresentada a seguir.

2

O Jogo da Velha

O Jogo da Velha foi desenvolvido em Python e possui a interface feita em Qt, de maniera bem simples, que permite a escolha de começar depois, assim como selecionar o nível de dificultade. Caso o usuário queira começar primeiro, apenas precisa marcar o local desejado.

Figura 1-Tela inicial do Jogo

Após o usuário efetuar a sua jogada, analisamos a profundidade, que se refere a um nível de dificuldade e para cada opção a resposta do jogo será diferente, sendo permitido as opções de 0 até 3 (4 níveis de dificuldade), e esse controle é realizado dentro do método da classe *game*, no arquivo *game.py* como está apresentado na figura abaixo.

```
class Game:
    def my_time(self, table, level):
        board_local = Board(table)
        tabela = self.formatt_talble(table)

        if(type(board_local.where_i_win()) == list and int(level) >= 1 ):
            line, colm = board_local.where_i_win()
            return [line, colm]
        elif(type(board_local.where_i_lose()) == list and int(level) >= 2):
            line, colm = board_local.where_i_lose()
            return [line, colm]
        elif(type(board_local.jogada_1()) == list and int(level) >= 3):
            line, colm = board_local.jogada_1()
            return [line, colm]
        else:
            auxTree = Tree(tabela,0)
            auxTree.build_min_max_with_depth()
            line, colm = auxTree.wich_one_is_the_best()
            return [line, colm]

```

Figura 2-Classe Game e seu método my_time

Ao solicitar que o computador faça sua jogada, é repassado para a classe *Game* o cenário do jogo (tabuleiro) e a pronfundidade (dificuldade).

Os niveis de dificuldade são realizado com as verificações do metodo *where_i_win*, o qual verifica se há possibilidades do computador ganhar a partida, e *where_i_lose* onde verifica se há possibilidades do usuário ganhar a partida.

A *jogada_1* verifica uma jogada clássica do Jogo da Velha, e caso o usuário esteja tentando fazê-la, o computador irá responder, para impedir que a jogada seja realizada.

Já o último é a realização do algoritimo do Minmax, o qual está com profundidade de busca única igual a três.

A relação dos níveis de dificuldade com o jogo, é que caso seja escolhida alguma profundidade, deverá ocorrer nenhuma, parcial ou todas as verificações, antes de realizar a tomada de decisão pelo Minmax.

- Dificuldade 0:
    Algoritimo Minmax;
- Dificuldade 1:
    Possibilidade do computador ganhar;
    Algoritimo Minmax;
- Dificuldade 2:
    Possibilidade do computador ganhar;
    Possibilidade do usuário ganhar;
    Algoritimo Minmax;
- Dificuldade 3:
    Possibilidade do computador ganhar;
    Possibilidade do usuário ganhar;
    Usuário está realizando Jogada Classica;
    Algoritimo Minmax;

3

Jogando

Fazendo testes para ver como se comporta o jogo na prática, o mesmo mostra-se interssante e consegue surpreender o usuário em momentos de descuidos, porém o jogo com o nível de profundidade (dificuldade) zero se mostra muito ingênuo, onde facilmente o usuário consegue vencer, tendo em vista que apenas o algoritimo do Minmax está sendo utilizado para tomada de decisão.

Figura 3-Usuário vencendo na dificuldade menor

Já no segundo teste aumentamos o nível de dificuldade para um e vemos uma melhora, onde ele consegue tomar a decisão de ganhar o jogo antes de fazer qualquer jogada, e apesar disso o computador ainda é um adversário fácil de se vencer.

Figura 4-Computador consegue vencer na dificuldade dois.

O Terceiro teste subimos o nível de dificuldade para dois, e podemos perceber que o computador se torna um bom adversário, conseguindo em certas cituações impedir que o adversário vença.

Figura 5-Computador é capaz de impedir vitória do adversário

E no último teste colocamos a dificuldade três e apenas conseguimos ganhar executando uma segunda jogada classica, já que a primeira nesse nível de profundidade é defendida pelo computador.

Figura 6-Computador impedindo jogada clássica

Figura 7-Computador não consegue impedir jogada clássica 2

Conclusão

Podemos observar que o programa se comporta bem e que pode ser divertido para o usuário. O método de tomada de decisão do Minmax consegue ser bem efiencete porem não imbativel, mas que a combinação entre a algoritimo do Minmax e algumas regras podem tornar o computador imbátivel.

Referências

Wikipedia, Jogo da Velha. Disponível em:<https://pt.wikipedia.org/wiki/Jogo_da_velha>
Acesso em: 01 de Abril de 2018

Python, Documentação Python3. Disponível em:<https://www.python.org/>
Acesso em: 25 de Março de 2018

Wikipedia, Minmax. Disponível em:<https://pt.wikipedia.org/wiki/Minimax>
Acesso em: 28 de Março de 2018

Henrique Vianna, Fundamentos de IA. Disponível em:<http://henriquevianna.com/code/ia/jogo-da-velha.html>
Acesso em: 23 de Março de 2018

QT, Documentação. Disponível em:<https://www.qt.io/qt-features-libraries-apis-tools-and-ide/>
Acesso em: 25 de Março de 2018

Wikihow, 3 Formas de ganhar no Jogo da Velha. Disponível em:
<https://pt.wikihow.com/Ganhar-no-Jogo-da-Velha>
Acesso em: 02 de Abril de 2018