UNIVERSIDADE DE ÉVORA

Escola de Ciências e Tecnologias

Engenharia Informática

Inteligência Artificial

2019/2020

- Jogos de dois jogadores -

- Jogos com informação completa determinísticos -

Docentes: Paulo Quaresma

Discentes: Sarah Simon Luz – 38116

Ana Ferro – 39872

27 de Abril de 2020

1.

(a) A estrutura de dados escolhida para representar os estados do jogo foi:

```
%estado_inicial(Lista de tuplos)
%Tuplo composto por uma posição,p(X,Y), e a jogada para essa posição
%Jogada simbolizada por:
%   x -> jogador x
%   o -> jogador o (computador)
%   v -> espaço vazio
estado_inicial([(p(1,1),o), (p(1,2),v), (p(1,3),v), (p(1,4),v), (p(1,5),v),
                (p(2,1),x), (p(2,2),o), (p(2,3),x), (p(2,4),v), (p(2,5),v),
                (p(3,1),x), (p(3,2),o), (p(3,3),x), (p(3,4),o), (p(3,5),x),
                (p(4,1),o), (p(4,2),x), (p(4,3),o), (p(4,4),o), (p(4,5),x)]).
```

Estrutura que é representada por uma lista de tuplos onde é guardada uma posição do tabuleiro e a jogada para essa posição. Jogada essa que pode assumir os caracteres:

"x"- representando o jogador;

"o"- representando a jogada do computador;

"v"- representando uma posição sem nenhuma jogada.

(b) O estado é terminal quando se verifica a existência de três x's ou o's seguidos numa linha, coluna ou diagonal qualquer, ou em último caso, empate quando todas as posições estão preenchidas e não houve vencedores.

```
terminal(E):-
    linhas(E);
    colunas(E);
    diagonais(E);
    empate(E).
```

(c) Para definir a função utilidade utilizámos o seguinte predicado:

```
valor(E,-1):-
    (linhas(E); colunas(E); diagonais(E)),
    vencedor(o), !.

valor(E,0):-
    empate(E), !.

valor(E,1):-
    (linhas(E); colunas(E); diagonais(E)),
    vencedor(x), !.
```

Esta função verifica a profundidade na árvore de pesquisa. Esta devolve -1, 0 ou 1, representando respetivamente, que perde, empata e ganha.

(e) Por demorar demasiado tempo partimos deste estado:

```
o | v | v | v | v |
x | o | x | v | v |
x | o | x | o | x |
o | x | o | o | x |
```

Média dos resultados

Minimax:

-Nº posições preenchidas: 14

-Tempo: 13s

-Nós: 66

α-β:

-Nº posições preenchidas:14

-Tempo:0s

-Nós:33

Concluímos que a pesquisa minimax com corte alfa-beta é o algoritmo mais ótimo pois leva menos tempo a efetuar a mesma jogada.

(f) Para o jogo suportar a existência de um agente inteligente criamos um ciclo de alternância entre o jogador e o computador até se verificar o estado terminal, ou seja, um dos jogadores ter alcançado a vitória ou o empate.

```
ciclo(_,_,E):-
    (linhas(E);colunas(E);diagonais(E)),
    tabuleiro(E),nl,
    write('Vencedor: '),
    vencedor(J),
    write(J), !.

ciclo(_,_,E):-
    empate(E),
    tabuleiro(E),nl,
    write('Empate'), !.
```

```
ciclo('min', o, E):-
    tabuleiro(E),nl,
    statistics(real_time,[Ti,_]),
    minimax_decidir(E,Op),
    statistics(real_time,[Tf,_]), T is Tf-Ti,
    write('Tempo: '),
    write(T), nl,
    write('Nós: '),
    nos(N),
    write(N),nl,
    write('Jogada computador: '),
    write(Op),nl,
    inicia_contador,
    oper(E,o,Op,Es),
    ciclo('min',x,Es).

ciclo('min',x,E):-
    tabuleiro(E),nl,
    write('Coluna:'),nl,
    read(X),nl,
    coluna(X,Op),
    oper(E, x, Op, Es),
    ciclo('min',o, Es).

ciclo('alf',o,E):-
    tabuleiro(E),nl,
    statistics(real_time,[Ti,_]),
    alfabeta(E,Op),
    statistics(real_time,[Tf,_]), T is Tf-Ti,
    write('Tempo: '),
    write(T), nl,
    write('Nós: '),
    nos(N),
    write(N),nl,
    write('Jogada computador:'),
    write(Op),nl,
    inicia_contador,
    oper(E,o,Op,Es),
    ciclo('alf',x,Es).
```

Para correr o trabalho:

1. ?- consult(main).
2. ?- main.
3. Escolher uma opção do menu

```
                MENU
1 - Jogar 3 em linha (minimax)
2 - Jogar 3 em linha (alfabeta)
3 - Sair
Opção:
```

4. O jogador (agente inteligente) é sempre o primeiro a jogar. Será pedido para introduzir a coluna da jogada
5. O jogo termina quando alguém ganhar ou empatar