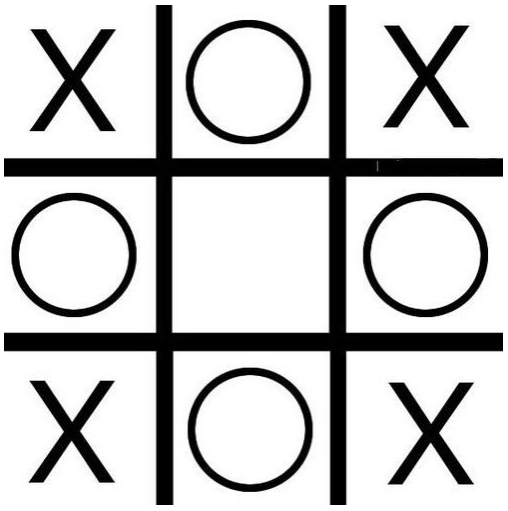
UNIVERSIDADE DE ÉVORA

INTELIGÊNCIA ARTIFICIAL



Jogos de dois jogadores- Jogos com informação completa determinísticos

Docente: Irene Pimenta Rodrigues
Discentes: José Albino nº 32096
Raquel Gomes nº 31523

Licenciatura em Engenharia Informática
Semestre Par
2016/2017

INTRODUÇÃO

Este terceiro trabalho prático foi realizado no âmbito da cadeira de Inteligência Artificial, lecionada pela docente Irene Pimenta Rodrigues.

Em relação ao objetivo deste trabalho pretendia-se a implementação do Jogo do Galo e de outro jogo de 2 jogadores à nossa escolha. Ambos os jogos tinham que cumprir um determinado número de regras e restrições para correrem com os algoritmos dados pela docente nas aulas práticas.

Além do Jogo do Galo, o segundo jogo escolhido para implementar foi o Quatro em Linha.

JOGO DO GALO

a. **Estrutura de dados para representar os estado do jogo.**

A estrutura escolhida para representar os estados do jogo “Jogo do Galo” foi uma lista.

```
%(lista com a posicao, ultima peca jogada)
estado_inicial(([
        (p(1,1), _),
        (p(1,2), _),
        (p(1,3), _),
        (p(2,1), _),
        (p(2,2), _),
        (p(2,3), _),
        (p(3,1), _),
        (p(3,2), _),
        (p(3,3), _)],_)).
```

b. **Defina o predicado terminal(estado) que sucede quando o estado e terminal.**

O predicado terminal(estado) vai ser o seguinte:

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

c. **Defina uma função de utilidade que para um estado terminal que deve retornar o valor do estado (ex: -1 perde, 0 empata, 1 ganha).**

Nesta função de utilidade para um estado terminal, se o tabuleiro de jogo já estiver todo preenchido e não ocorrer nenhum caso em que estejam três peças iguais em linha, sabemos que é um caso de empate. Caso o tabuleiro de jogo já esteja preenchido e cada jogador já tenha jogado três vezes, e um deles tenha posto três peças iguais em linha/coluna/diagonal, esse jogador é então o que ganha e é impresso no ecrã.

```
valor((E, _), 1, _):-
        (linhas(E);colunas(E);diagonais(E)),
        ganha(o),
        !. %cut

valor((E, _), -1, _):-
        (linhas(E);colunas(E);diagonais(E)),
        ganha(x),
        !.

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

d. **Use a implementação da pesquisa minimax dada na aula prática para escolher a melhor jogada num estado.**

A implementação da pesquisa minimax dada na aula prática encontra-se anexada no ficheiro minmax.pl.

e. **Implemente a pesquisa Alfa-Beta e compare os resultados (tempo e espaço).**

A pesquisa Alfa-Beta está definida no ficheiro anexado alfabeta.pl.

f. **Defina uma função de avaliação que estime o valor de cada estado do jogo use os dois algoritmos anteriores com corte em profundidade e compare os resultados (tempo e espaço).**

A função de avaliação feita para este jogo foi fazer o calculo de uma peça isolada somando ao número de duas peças juntas, fazer a mesma soma para o oponente e subtrair um valor ao outro. Quando somamos um número de duas peças damos um peso extra a esse número e multiplicamos por 2.

```
% avalia(Estado, Tipo_peca, Avaliacao)
%É dado um estado, o tipo de peca (x ou o), retorna em C o valor da avaliacao

func_avalia((E,J), Val,_):-
        inverteJogada(J, J2),
        avalia(E,J2,Val).

avalia(E, J, Val):-
        find_all_1peca(E, J, V1),
        find_all_2pecas(E, J, V2),
        Val1 is V1+(3*V2),
        inverteJogada(J, J2),
        find_all_1peca(E, J2, V3),
        find_all_2pecas(E, J2, V4),
        Val2 is V3+(3*V4),
        Val is (Val1-Val2).
```

g. Implemente um agente inteligente que joga o jogo do galo.

O agente inteligente que joga o “Jogo do Galo”:

```
%Pergunta ao jogador qual a linha  ou a coluna em que quer jogar
%read - lê o próximo termo de input e unifica-o como termo
%Agente inteligente

ciclo_jogo('j',(E,J)):-
        print_(E),
        nl,
        write('Escreva a linha da posicao onde deseja jogar: '),
        read(X),
        write('Escreva a coluna da posicao onde deseja jogar: '),
        read(Y),
        inverteJogada(J,J1),
        op1((E,J),
        insere(p(X,Y),J1),Es),
        cilo_jogo('c',Es).
```

Este agente, utilizando o algoritmo “minimax” ou o algoritmo “Alfa-Beta”, preenche o tabuleiro de modo a que impeça que o jogador humano ganhe.

h. **Apresente uma tabela com o número de nós expandidos para diferentes estados do jogo (10 no mínimo) com os vários algoritmos.**

O número de nós expandidos para diferentes estados do jogo com os dois algoritmos são:

Minimax: **Alfa-Beta:**

QUATRO EM LINHA

1 - Estrutura de dados para representar os estado do jogo.

A estrutura escolhida para representar os estados do jogo foi uma lista.

```
%(lista com a posicao, ultima peca jogada)
estado_inicial(([(p(1,1), _),(p(1,2), _),(p(1,3), _),(p(1,4), _),(p(1,5), _),(p(1,6), _),(p(1,7), _),
                 (p(2,1), _),(p(2,2), _),(p(2,3), _),(p(2,4), _),(p(2,5), _),(p(2,6), _),(p(2,7), _),
                 (p(3,1), _),(p(3,2), _),(p(3,3), _),(p(3,4), _),(p(3,5), _),(p(3,6), _),(p(3,7), _),
                 (p(4,1), _),(p(4,2), _),(p(4,3), _),(p(4,4), _),(p(4,5), _),(p(4,6), _),(p(4,7), _),
                 (p(5,1), _),(p(5,2), _),(p(5,3), _),(p(5,4), _),(p(5,5), _),(p(5,6), _),(p(5,7), _),
                 (p(6,1), _),(p(6,2), _),(p(6,3), _),(p(6,4), _),(p(6,5), _),(p(6,6), _),(p(6,7), _)], _)).
```

2 - Defina o predicado terminal(estado) que sucede quando o estado e terminal.

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

3 - Defina uma função de utilidade que para um estado terminal que deve retornar o valor do estado (ex: -1 perde, 0 empata, 1 ganha).

Nesta função de utilidade para um estado terminal, se o tabuleiro de jogo já estiver todo preenchido e não ocorrer nenhum caso em que estejam quatro peças iguais em linha/coluna/diagonal, sabemos que é um caso de empate. Caso o tabuleiro de jogo já esteja ou não preenchido e cada jogador já tenha jogado pelo menos quatro vezes, e um deles tenha posto quatro peças iguais em linha/coluna/diagonal, esse jogador é então o que ganha e é impresso no ecrã.

```
valor((E, _), 100, _):-
        (linhas(E);colunas(E);diagonais(E)),
        ganhador(o),
        !.

valor((E, _), -100, _):-
        (linhas(E);colunas(E);diagonais(E)),
        ganhador(x),
        !.

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

4 - Implemente um agente inteligente que joga o jogo do galo.

5 - Apresente uma tabela com o número de nós expandidos para diferentes estados do jogo (10 no mínimo) com os vários algoritmos.