Relatório do 3º Trabalho Prático

Inteligência Artificial

Universidade de Évora
Engenharia Informática 2020/2021

Docente: Irene Rodrigues
Discentes: Leonardo Catarro, 43025
Diogo Solipa, 43071

Desenvolvimento

Exercício 1

a) Representação do estado:

e(a,b,c,d) :

- a,b,c,d são o número de pauzinhos por coluna. Para facilitar o trabalho consideramos que o jogo é constituído apenas por 4 colunas. Tendo 1,3,5,7 pauzinhos cada uma, respetivamente. Foi usado como estrutura para representar o estado um "tuplo" de 4 elementos.

```
%
% Estado inicial
%
estado_inicial(e(7,5,3,1)).
```

b)

```
%
% Estado terminal
%
terminal(e(0,0,0,0)).
```

c)

```
%
% Funcao utilidade(1 se primeiro jogador ganha ; -1 se perde)
%
valor2(X, Res) :- X='x' -> Res is 1 ; Res is -1.
printValor(X) :- X=1 -> write('Jogador Ganhou!') ; write('Jogador Adversário Ganhou!').
```

d) e) f) g) h)

Derivado à inúmera presença de bugs e problemas como o desenvolvimento com a linguagem declarativa Prolog, não conseguimos por o programa a funcionar com o Minimax, nem com o Alfabeta. Ainda assim temos presente nos ficheiros do trabalho este algoritmos implementados(fornecidos em âmbito de aula, pela professora).

Exercício 2

Para o outro jogo de 2 jogadores, escolhemos uma versão simplificada do 4 em linha, ao qual chamamos o 3 em Linha.

a)

```
%tabuleiro de jogo
/*
        (1,1)(2,1)(3,1)
        (1,2)(2,2)(3,2)
        (1,3)(2,3)(3,3)
        (1,4)(2,4)(3,4)
*/

%estado inicial
estado_inicial([[v,v,x], [v,o,o], [o,x,o], [x,o,x]]).
```

b)

```
%COLUNAS
% 1ª coluna
colunas([[X,X,X],_,_,_], X):- X \= v.
colunas([[X,X,X],_,_,_], X):- X \= v.
% 2ª coluna
colunas([_,[X,X,X],_,_], X):- X \= v.
colunas([_,[X,X,X],_,_], X):- X \= v.
% 3ª coluna
colunas([_,_,[X,X,X],_], X):- X \= v.
colunas([_,_,[X,X,X],_], X):- X \= v.
% 4ª coluna
colunas([_,_,_,[X,X,X]], X):- X \= v.
colunas([_,_,_,[X,X,X]], X):- X \= v.

%LINHAS
% 1ª linha
linhas([[X|_],[X|_],[X|_],_], X):- X \= v.
linhas([_,[X|_],[X|_],[X|_]], X):- X \= v.
% 2ª linha
linhas([[_,X|_],[_,X|_],[_,X|_],_], X):- X \= v.
linhas([_,[_,X|_],[_,X|_],[_,X|_]], X):- X \= v.
% 3ª linha
linhas([[_,_,X],[_,_,X],[_,_,X],_], X):- X \= v.
linhas([_,[_,_,X],[_,_,X],[_,_,X]], X):- X \= v.

%DIAGONAIS
% Diagonais Ascendentes
diagonais([[_,_,X],[_,X,_],[X,_,_],_], X):- X \= v.
diagonais([_,[_,_,X],[_,X,_],[X,_,_]], X):- X \= v.
% Diagonais Descendentes
diagonais([[X,_,_],[_,X,_],[_,_,X],_], X):- X \= v.
diagonais([_,[X,_,_],[_,X,_],[_,_,X]], X):- X \= v.
```

```
% Predicado em que o tabuleiro ta cheio
cheio([C1, C2, C3, C4]):-
    append(C1, C2, C12),
    append(C3, C4, C34),
    append(C12, C34, FB),
    \+ member(v, FB).
```

```
terminal(F):- linhas(F,_).
terminal(F):- colunas(F,_).
terminal(F):- diagonais(F,_).
terminal(F):- cheio(F).
```

c)

```
% Funcao de utilidade, retorna o valor dos estados terminais 1 ganha -1 perde
valor(F, 1):- linhas(F, x).
valor(F, 1):- colunas(F, x).
valor(F, 1):- diagonais(F, x).
valor(F, -1):- linhas(F, o).
valor(F, -1):- colunas(F, o).
valor(F, -1):- diagonais(F, o).
valor(_,0).
```

d)

Neste trabalho não conseguimos implementar o agente inteligente, porém ainda assim, conseguimos ter o minimax funcionar retornando a melhorar jogada possível para um estado. O alfabeta ficou com uns bugs, não sendo possível a sua execução.

```
Melhor Jogada para o estado: [[v,v,x],[v,o,o],[o,x,o],[x,o,x]]
joga(1,2)
```

e)

Estado	Nós expandidos Minimax	Nós expandidos Alfabeta
[[v,v,x], [v,o,o], [o,x,o], [x,o,x]]).	8	-
[[v,v,v], [v,o,o], [o,x,o], [x,o,x]]	13	-
[[v,v,v], [v,v,o], [o,x,o], [x,o,x]]	33	-
[[v,v,v], [v,v,v], [o,x,o], [x,o,x]]	68	-
[[v,v,v], [v,v,v], [o,x,o], [x,o,v]]	447	-
[[v,v,v], [v,v,v], [o,x,o], [x,v,v]]	1748	-

[[v,v,v], [v,v,v], [o,x,o], [v,v,v]]	5247	-
[[v,v,v], [v,v,v], [o,x,v], [v,v,v]]	50929	-
[[v,v,v], [v,v,v], [o,v,v], [v,v,v]]	276923	-
[[v,v,v], [v,v,v], [v,v,v], [v,v,v]]	1107696	-

Conclusão

Com este trabalho conseguimos por em prática todo o conhecimento lecionado nas aulas teóricas e práticas da disciplina relativamente a jogos de 2 jogadores.
Embora tenhamos tido dificuldades na resolução do 1º exercício, visto que não percebemos bem como lidar como corrigir os bugs ao nível do Prolog e na aplicação dos algoritmos ao jogo do Nim, no geral consideramos que conseguimos aplicar o conhecimento adquirido.
A experiência de desenvolver jogos de 2 jogadores em Prolog, foi bastante desafiante, mas também gratificante e interessante.