



UNIVERSIDADE DE ÉVORA

Engenharia Informática
Inteligência Artificial
Terceiro Trabalho



Autor:

Gabriel Charrua 32457

Docente:

Irene Pimenta Rodrigues

2015/2016

Introdução

Antes de mais, a realização deste trabalho encontra-se inserida na cadeira de Inteligência Artificial mais especificamente na componente prática. A realização deste trabalho foi proposta pela docente Irene Pimenta Rodrigues. Em relação à cadeira, encontra-se inserida na Licenciatura em Engenharia Informática da Universidade de Évora, 6o semestre.

Em relação ao propósito deste trabalho, pretende-se fazer o “Jogo do Galo” e um jogo à escolha, neste caso, o “Quatro em Linha. Ambos os jogos vão ter por objectivo cumprir um determinado conjunto de regras e restrições para que corram com os algoritmos dados nas aulas teóricas e nas aulas práticas.

1. Escolha uma estrutura de dados para representar os estados dos dois jogos.

R: A estrutura utilizada para representar o estado do jogo é representado por um tuplo com uma lista de posições do tabuleiro, onde cada posição contém uma referencia(um valor não instanciado), um caractere "x" ou "o" que indica uma jogada e a última peça que foi jogada. Em baixo podem ver-se as representações dos estados iniciais para o "Jogo do Galo":

```
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), _)], _)).  
  
estado_inicial(([g,g,g,p,p,p,p,g,p,p,g,g,g,p,g,g,g,p,p,p], _, _, _)).
```

e para o "Quatro em Linha":

```
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 é terminal para cada jogo.

R: Um estado é terminal se há uma linha, coluna ou diagonal completa, seja ela com x's ou com o's, ou no ultimo dos casos quando estão todas as posições preenchidas e não há ganhadores(empate).

```
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), para cada jogo.

R: A função de utilidade verifica a profundidade na árvore de pesquisa e os casos em que o estado é terminal, com a excepção do empate. Os valores devolvidos pela mesma podem ser 1, 0 ou -1 sendo que 0 representa empate, 1 ganha e -1 perde.

```
valor((E, _), 1, _):- (linhas(E); colunas(E); diagonais(E)), ganhador(o), !.  
valor((E, _), -1, _):- (linhas(E); colunas(E); diagonais(E)), ganhador(x), !.  
valor((E, _), 0, _):- empate(E), !.
```

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

R: Foi utilizada a implementação dada pela professora nas aulas para a realização do trabalho (escolher a melhor jogada num estado)

```
1 g(Jogo):- [Jogo], estado_inicial(Ei), minimax_decidir(Ei,Op),nl,  
2 write(Op),nl.  
3  
4 % decide qual é a melhor jogada num estado do jogo  
5 % minimax_decidir(Estado, MelhorJogada)  
6  
7 % se é estado terminal não há jogada  
8 minimax_decidir(Ei,terminou):- terminal(Ei).  
9  
10 %Para cada estado sucessor de Ei calcula o valor minimax do estado  
11 %Opf é o operador (jogada) que tem maior valor  
12  
13 minimax_decidir(Ei,Opf):-  
14     findall(Es-Op, opl(Ei,Op,Es),L),  
15     length(L,S),  
16     incMais(S),  
17     findall(Vc-Op,(member(E-Op,L), minimax_valor(E,Vc,l)),LI),  
18     escolhe_max(LI,Opf).  
19  
20 % se um estado é terminal o valor é dado pela função de utilidade  
21 %minimax_valor(Ei,Val,P):- terminal(Ei), valor(Ei,Val,P).  
22 minimax_valor(Ei,Val,P):- terminal(Ei), !, valor(Ei,Val,P).  
23  
24  
25 %Se o estado não é terminal o valor é:  
26 % -se a profundidade é par, o maior valor dos sucessores de Ei  
27 % -se a profundidade é impar o menor valor dos sucessores de Ei  
28 minimax_valor(Ei,Val,P):-  
29     findall(Es,opl(Ei,_,Es),L),  
30     length(L,S),  
31     incMais(S),  
32     PI is P+1,  
33     findall(ValI,(member(E,L),minimax_valor(E,ValI,PI)),V),  
34     seleciona_valor(V,P,Val).  
35  
36  
37 % Se a profundidade (P) é par, retorna em Val o maximo de V  
38 seleciona_valor(V,P,Val):- X is P mod 2, X=0,!, maximo(V,Val).  
39  
40 % Senão retorna em Val o minimo de V  
41 seleciona_valor(V,_,Val):- minimo(V,Val).  
42  
43  
44 maximo([A|R],Val):- maximo(R,A,Val).  
45  
46 maximo([],A,A).  
47 maximo([A|R],X,Val):- A < X,!, maximo(R,X,Val).  
48 maximo([A|R],_,Val):- maximo(R,A,Val).  
49  
50  
51 escolhe_max([A|R],Val):- escolhe_max(R,A,Val).  
52  
53 escolhe_max([],_Op,Op).  
54 escolhe_max([A-_|R],X-Op,Val):- A < X,!, escolhe_max(R,X-Op,Val).  
55 escolhe_max([A|R],_,Val):- escolhe_max(R,A,Val).  
56  
57 minimo([A|R],Val):- minimo(R,A,Val).  
58  
59 minimo([],A,A).  
60 minimo([A|R],X,Val):- A > X,!, minimo(R,X,Val).  
61 minimo([A|R],_,Val):- minimo(R,A,Val).  
62
```

5. Implemente a pesquisa Alfa-Beta e compare os resultados (tempo e espaço) em exemplos com os dois jogos

R: Através dos resultados observados, pode-se concluir que o minimax, apesar de demorar mais tempo a efectuar as jogadas, faz sempre uma jogada óptima, enquanto que o corte α - β não faz a melhor jogada óptima mas leva menos tempo a efectuar a mesma. Foram realizadas 10 observações de tempo e nº de nós para jogadas com 1, 3 e 5 posições preenchidas. A média dos resultados obtidos pode ser observada através da tabela 1 para o algoritmo minimax e na tabela 2 para o corte α - β .

Tabela1 (minimax):

Nº Posições Preenchidas	Tempo (ms)	Nº de Nós
1	4324	59704
3	81.2	1109
5	6	51

Tabela 2 (α - β):

Nº Posições Preenchidas	Tempo (ms)	Nº de Nós
1	29	337
3	5	18
5	3	9

6. 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), com exemplos dos dois jogos.

R: A função de avaliação feita para este jogo foi fazer o cálculo do número de 1 peça isolada, somando ao número de 2 peças juntas, fazer a mesma soma para o oponente e subtrair um valor ao outro. Quando se soma o numero de 2 peças, dá-se um valor extra a este numero, multiplicando-o por 2.

```
func_aval((E,J), Val,_) :-
    inverteJog(J, J2),
    aval(E, J2, Val).

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

7. Implemente um agente inteligente que joga o jogo que escolheu usando a pesquisa definida na alínea anterior.

R: Implementou-se um agente inteligente que joga contra o jogador até que se verifique um estado terminal, mas apenas para o jogo do galo, não sendo possível para o 4 em Linha.

```
171  inverteJog('x','o').
172  inverteJog('o','x').
173
174
175  ciclo_jogada(_, (E,J)):- (linhas(E);colunas(E);diagonais(E)), print_(E),write('Vencedor: '),write(J),!.
176  ciclo_jogada(_, (E,_)):- empate(E), print_(E),write('Empate!'),nl,!.
177
178  ciclo_jogada('c', (E,J)):-
179      print_(E),
180      nl,statistics(real_time,[Ti,_]),
181      minimax_decidir((E,J),Op),
182      statistics(real_time,[Tf,_]), T is Tf-Ti,
183      nl,
184      write('Tempo: '(T)),
185      nl,
186      n(N),
187      write('Numero de nos: '(N)),
188      initInc,
189      nl,
190      write(Op),
191      nl,
192      nl,
193      opl((E,J),Op,Es),
194      ciclo_jogada('j',Es).
195
196  ciclo_jogada('j', (E,J)):-
197      print_(E),
198      nl,
199      write('Escreva a linha da posicao onde deseja jogar: '),
200      read(X),
201      write('Escreva a coluna da posicao onde deseja jogar: '),
202      read(Y),
203      inverteJog(J,JI),
204      opl((E,J),insere(p(X,Y),JI),Es),
205      ciclo_jogada('c',Es).
206
207  print_(E):-
208      print_linhas(E).
209
210  print_linhas(E):-
211      write(' '),
212      print_linha(E, 1, 1),
213      write(' '),
214      write_line(1, 3),
215      write(' '),
216      print_linha(E, 2, 1),
217      write(' '),
218      write_line(1, 3),
219      write(' '),
220      print_linha(E, 3, 1),
221      write('\n\n').
222
```

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

R: Não foi possível, para um mesmo jogo, expandir os nós com ambos os algoritmos, uma vez que o ciclo de jogo contém alguns bugs.