
T3-IA



Gustavo Oliveira nº46395, Berke Balci nº64498, Semiha Çetintaş nº64751

Évora, 25 de junho de 2025



1 - Considere a seguinte variante do jogo do galo.

Joga-se num tabuleiro de 3 X 3. Cada jogador pode colocar uma peças: 'S' ou 'O' no tabuleiro. O primeiro jogador a obter 'SOS' numa linha, coluna ou diagonal ganha.

A - Escolha uma estrutura de dados para representar os estado do jogo.

```
1 estado_inicial(e([v,v,v,v,v,v,v,v], s)).
```

B - Defina o predicado terminal (estado) que sucede quando o estado é terminal.

```
1 terminal(e([S,0,S,_,_,_,_,_],_)):- S==s; 0 == o.
2 terminal(e([_,_,S,0,S,_,_,_],_)):- S==s; 0 == o.
3 terminal(e([_,_,_,_,_,S,0,S],_)):- S==s; 0 == o.
4
5 terminal(e([S,_,_,0,_,_,S,_,_],_)):- S==s; 0 == o.
6 terminal(e([_,S,_,_,0,_,_,S,_,_],_)):- S==s; 0 == o.
7 terminal(e([_,_,S,_,_,0,_,_,S,_,_],_)):- S==s; 0 == o.
8
9 terminal(e([S,_,_,_,0,_,_,_,S],_)):- S==s; 0 == o.
10 terminal(e([_,_,S,_,_,0,_,_,_,S],_)):- S==s; 0 == o.
11
12 terminal(e(L,_)):- \+ member(v,L).
```

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

```
1 valor(e(L,_),O,P):- \+ member(v,L),!.
2
3 valor(E,V,P):-terminal(E),
4 X is P mod 2,
5 (X== 1,V=1;X==0,V= -1).
```

D - Use a implementação da pesquisa minimax dada na aula prática para escolher a melhor jogada num estado. Teste a sua descrição do jogo com vários estados.

*Exemplo-1:

```
1 estado_inicial(e([v,v,v,s,o,v,v,v], s)).
```

```
1 | ?- g(sos).
2 6,s
```

*Exemplo-2:

```
1 estado_inicial(e([s,v,v,v,o,v,v,v], s)).
```

```
1 | ?- g(sos).
2 9,s
```

*Exemplo-3:

```
1 estado_inicial(e([s,v,v,v,v,v,v,s], s)).
```

```
1 | ?- g(sos).
2 5,o
```

*Exemplo-4:

```
1 estado_inicial(e([s,s,s,v,v,s,v,v], s)).
```



```
1 | ?- g(sos).  
2   5,o
```

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

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).

```
1 % Predicado principal: soma os valores heur sticos dos padr es  
2 avalia(e(L, _), V) :-  
3     findall(V1, aval(L, V1), Vs),  
4     soma(Vs, V).  
5  
6 % Padr es a favor (potenciais SOS)  
7 aval(L, 1) :- nth1(1,L,s), nth1(2,L,o), nth1(3,L,v).  
8 aval(L, 1) :- nth1(1,L,s), nth1(3,L,o), nth1(2,L,v).  
9 aval(L, 1) :- nth1(2,L,s), nth1(3,L,o), nth1(1,L,v).  
10  
11 aval(L, 1) :- nth1(1,L,s), nth1(5,L,o), nth1(9,L,v). % diagonal  
12 aval(L, 1) :- nth1(3,L,s), nth1(5,L,o), nth1(7,L,v). % diagonal  
13  
14 % Padr es contra (amea as do advers rio)  
15 aval(L, -1) :- nth1(1,L,o), nth1(2,L,s), nth1(3,L,v).  
16 aval(L, -1) :- nth1(1,L,o), nth1(3,L,s), nth1(2,L,v).  
17 aval(L, -1) :- nth1(2,L,o), nth1(3,L,s), nth1(1,L,v).  
18  
19 aval(L, -1) :- nth1(1,L,o), nth1(5,L,s), nth1(9,L,v).  
20 aval(L, -1) :- nth1(3,L,o), nth1(5,L,s), nth1(7,L,v).  
21  
22 soma([], 0).  
23 soma([X|Xs], R) :-  
24     soma(Xs, R1),  
25     R is X + R1.
```

G - Implemente um agente inteligente que joga o SOS: 1 - Joga uma peça, atualiza e mostra o tabuleiro 2 - L a jogada do adversário e actualiza e mostra o tabuleiro, volta a 1 até o jogo terminar.

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

Tabela 1: Número de nós expandidos minimax

Estado #	Casas livres (k)	Nós expandidos (2^k)	Tabuleiro (exemplo)
1	9	512	[v,v,v,v,v,v,v,v,v]
2	8	256	[x,v,v,v,v,v,v,v]
3	7	128	[x,x,v,v,v,v,v,v]
4	6	64	[x,x,x,v,v,v,v,v]
5	5	32	[x,x,x,x,v,v,v,v]
6	4	16	[x,x,x,x,x,v,v,v]
7	3	8	[x,x,x,x,x,x,v,v]
8	2	4	[x,x,x,x,x,x,x,v]
9	1	2	[x,x,x,x,x,x,x,x]
10	0	1 (terminal)	[x,x,x,x,x,x,x,x]

2 - Considere a seguinte variante do jogo do galo.

em que o jogo só termina quando já não há casas livres e ganha o jogador que fez mais 'SOS's.



A - Escolha uma estrutura de dados para representar os estado do jogo.

```
1 estado_inicial(e([v,v,v,v,v,v,v,v],s,p1,p2)).
```

B - Defina o predicado terminal(estado) que sucede quando um estado é terminal.

```
1 terminal(e(Tabuleiro, _, _, _)) :-  
2 \+ member(v, Tabuleiro).
```

C - Defina uma função de utilidade que para um estado terminal que deve retornar o valor do estado.

```
1 valor(e(_, _, P1, P2), 1) :- P1 > P2. % Vit ria de j1 (MAX)  
2 valor(e(_, _, P1, P2), -1) :- P2 > P1. % Vit ria de j2 (MIN)  
3 valor(e(_, _, P1, P2), 0) :- P1 == P2. % Empate
```