UNIVERSIDADE DE ÉVORA

3º Trabalho de Inteligência Artificial

Yaroslav Kolodiy `139859` Eduardo Medeiros `139873`

Abril, Ano Letivo 2019/2020

Inteligência Artificial

Prof. Paulo Quaresma

Índice

1 Respostas

1.1 Nota Prévia

Nota prévia:

O programa *3emlinha.pl* deve ser compilado juntamente com o código de *minimax.pl* ou de *afabeta.pl*.

Após compilado através de uma das combinações anteriores, o predicado joga/0 deve ser chamado para se iniciar o jogo. Logo de seguida é mostrado o tabuleiro (previamente preenchido com algumas peças) e o jogador deve fornecer a coluna onde pretende colocar a sua peça. Seguidamente será a vez do programa jogar. O processo repetir-se-á até que haja um empate ou uma vitória.

1.2 Respostas

(a) Um estado é representado por uma lista de listas onde cada lista interior representa uma linha do tabuleiro.

(b) Um estado é considerado terminal se nele existe uma sequência de três "x"ou três "o"seguidos, seja esta sequência em qualquer vertical, horizontal ou diagonal.

(c) A função definida foi valor/2 que recebe um estado terminal do tabuleiro e devolve:

- -1 caso seja uma derrota para o programa
- 0 em caso de empate
- 1 caso seja vitória do jogador

(d) Representado se compilados juntos os códigos presentes em 2.1 e em 2.2.

(e) Usando em ambos os casos o estado inicial presente em 2.1, os resultado, aproximados, em termos temporais são ± 0.5s e ± 94s para os algoritmos 2.3 e 2.2 respectivamente. Através de uma abordagem teórica, o algoritmo *alfabeta.pl*, num caso ótimo, visitará menos estados que o algoritmo *minimax.pl* pois este corta/descarta os que não terá necessidade de visitar quando uma decisão já tiver sido tomada.

(f) Ambas as implementações, tanto a combinação de *3emlinha.pl* (2.1) com *alfabeta.pl* (2.3), tanto como a *3emlinha.pl* (2.1) com *minimax.pl* (2.2), representam um agente inteligente.

2 Anexos

2.1 Anexo 2 - 3emlinha.pl

```
1  % cada posicao pode ter "x", "o" ou "v" (vazio)
2  %estado_inicial([[v,v,v,v,v],[v,v,v,v,v],[x,o,v,v,v],[o
       ,x,x,o,v]]).
3  estado_inicial([[v,v,v,v,v],[v,v,v,v,v],[x,o,x,v,v],[o,
       x,x,o,o]]).
4  %estado_inicial([[v,v,v,v,v],[v,v,v,v,v],[v,v,v,v,v],[v
       ,v,v,v,v]]).
5
6  terminal(G) :- linhas(G,_).
7  terminal(G) :- colunas(G,_).
8  terminal(G) :- diagonal(G,_).
9  terminal(G) :- cheio(G).
10
11 linhas([[X,X,X,_,_],_,_,_],X) :- X \= v.
12 linhas([[_,X,X,X,_],_,_,_],X) :- X \= v.
13 linhas([[_,_,X,X,X],_,_,_],X) :- X \= v.
14
15 linhas([_,[X,X,X,_,_],_,_],X) :- X \= v.
16 linhas([_,[_,X,X,X,_],_,_],X) :- X \= v.
17 linhas([_,[_,_,X,X,X],_,_],X) :- X \= v.
18
19 linhas([_,_,[X,X,X,_,_],_],X) :- X \= v.
20 linhas([_,_,[_,X,X,X,_],_],X) :- X \= v.
21 linhas([_,_,[_,_,X,X,X],_],X) :- X \= v.
22
23 linhas([_,_,_,[X,X,X,_,_]],X) :- X \= v.
24 linhas([_,_,_,[_,X,X,X,_]],X) :- X \= v.
25 linhas([_,_,_,[_,_,X,X,X]],X) :- X \= v.
26
27 colunas([[X,_,_,_,_],[X,_,_,_,_],[X,_,_,_,_], [_,_,_,_,
       _]],X) :- X \= v.
28 colunas([[_,_,_,_,_],[X,_,_,_,_],[X,_,_,_,_], [X,_,_,_,
       _]],X) :- X \= v.
29
30 colunas([[_,X,_,_,_],[_,X,_,_,_],[_,X,_,_,_], [_,_,_,_,
       _]],X) :- X \= v.
31 colunas([[_,_,_,_,_],[_,X,_,_,_],[_,X,_,_,_], [_,X,_,_,
```

```
      _]],X) :- X \= v.
32
33 colunas([[_,_,X,_,_],[_,_,X,_,_],[_,_,X,_,_], [_,_,_,_,_,
      _]],X) :- X \= v.
34 colunas([[_,_,_,_,_],[_,_,X,_,_],[_,_,X,_,_], [_,_,X,_,
      _]],X) :- X \= v.
35
36 colunas([[_,_,_,X,_],[_,_,_,X,_],[_,_,_,X,_], [_,_,_,_,_,
      _]],X) :- X \= v.
37 colunas([[_,_,_,_,_],[_,_,_,X,_],[_,_,_,X,_], [_,_,_,X,
      _]],X) :- X \= v.
38
39 colunas([[_,_,_,_,X],[_,_,_,_,X],[_,_,_,_,X], [_,_,_,_,_,
      _]],X) :- X \= v.
40 colunas([[_,_,_,_,_],[_,_,_,_,X],[_,_,_,_,X], [_,_,_,_,_,
      X]],X) :- X \= v.
41
42
43 diagonal([[X,_,_,_,_],[_,X,_,_,_],[_,_,X,_,_], [_,_,_,_,
      _]],X) :- X \= v.
44 diagonal([[_,_,_,_,_],[_,X,_,_,_],[_,_,X,_,_], [_,_,_,X
      _]],X) :- X \= v.
45 diagonal([[_,_,_,_,_],[X,_,_,_,_],[_,X,_,_,_], [_,_,X,_
      _]],X) :- X \= v.
46 diagonal([[_,X,_,_,_],[_,_,X,_,_],[_,_,_,X,_], [_,_,_,_,
      _]],X) :- X \= v.
47 diagonal([[_,_,_,_,_],[_,_,X,_,_],[_,_,_,X,_], [_,_,_,_,
      X]],X) :- X \= v.
48 diagonal([[_,_,X,_,_],[_,_,_,X,_],[_,_,_,_,X], [_,_,_,_,
      _]],X) :- X \= v.
49
50 diagonal([[_,_,X,_,_],[_,X,_,_,_],[X,_,_,_,_], [_,_,_,_,
      _]],X) :- X \= v.
51 diagonal([[_,_,_,_,_],[_,_,X,_,_],[_,X,_,_,_], [X,_,_,_,
      _]],X) :- X \= v.
52 diagonal([[_,_,_,X,_],[_,_,X,_,_],[_,X,_,_,_], [_,_,_,_,
      _]],X) :- X \= v.
53 diagonal([[_,_,_,_,X],[_,_,_,X,_],[_,_,X,_,_], [_,_,_,_,
      _]],X) :- X \= v.
54 diagonal([[_,_,_,_,_],[_,_,_,X,_],[_,_,X,_,_], [_,X,_,_,
      _]],X) :- X \= v.
```

```
55 diagonal([[_,_,_,_,_],[_,_,_,_,_,X],[_,_,_,_,X,_], [_,_,X,_
       ,_]],X) :- X \= v.
56
57 cheio([L1,L2,L3,L4]) :-
58      append(L1,L2, L12),
59      append(L12, L3, L123),
60      append(L123, L4, L1234),
61      \+ member(v, L1234).
62
63 %fun   o de utilidade, retorna o valor dos estados
       terminais, 1 ganha -1 perde
64 valor(G, 1) :- linhas(G,x).
65 valor(G, 1) :- colunas(G,x).
66 valor(G, 1) :- diagonal(G,x).
67 valor(G, -1) :- linhas(G,o).
68 valor(G, -1) :- colunas(G,o).
69 valor(G, -1) :- diagonal(G,o).
70 valor(_, 0).
71
72 % oper(estado,jogador,jogada,estado seguinte)
73 oper(E, J,joga(X,Y), En) :-
74          joga_vazio(E,J,X, Y, En).
75
76
77 joga_vazio([[v, C12, C13, C14, C15],
78              [C21, C22, C23, C24, C25],
79              [C31, C32, C33, C34, C35],
80              [C41, C42, C43, C44, C45]],
81             J,
82             1,1,
83             [[J, C12, C13, C14, C15],
84             [C21, C22, C23, C24, C25],
85             [C31, C32, C33, C34, C35],
86             [C41, C42, C43, C44, C45]]):- C21 \= v, C31
                   \= v, C41 \= v.
87
88 joga_vazio([[C11, C12, C13, C14, C15],
89              [v, C22, C23, C24, C25],
90              [C31, C32, C33, C34, C35],
91              [C41, C42, C43, C44, C45]],
92             J,
```

```
93              2,1,
94              [[C11, C12, C13, C14, C15],
95              [J, C22, C23, C24, C25],
96              [C31, C32, C33, C34, C35],
97              [C41, C42, C43, C44, C45]]):- C31 \= v,
                   C41 \= v.
98
99  joga_vazio([[C11, C12, C13, C14, C15],
100             [C21, C22, C23, C24, C25],
101             [v, C32, C33, C34, C35],
102             [C41, C42, C43, C44, C45]],
103             J,
104             3,1,
105             [[C11, C12, C13, C14, C15],
106             [C21, C22, C23, C24, C25],
107             [J, C32, C33, C34, C35],
108             [C41, C42, C43, C44, C45]]):- C41 \= v.
109
110 joga_vazio([[C11, C12, C13, C14, C15],
111             [C21, C22, C23, C24, C25],
112             [C31, C32, C33, C34, C35],
113             [v, C42, C43, C44, C45]],
114             J,
115             4,1,
116             [[C11, C12, C13, C14, C15],
117             [C21, C22, C23, C24, C25],
118             [C31, C32, C33, C34, C35],
119             [J, C42, C43, C44, C45]]).
120
121 joga_vazio([[C11, v, C13, C14, C15],
122             [C21, C22, C23, C24, C25],
123             [C31, C32, C33, C34, C35],
124             [C41, C42, C43, C44, C45]],
125             J,
126             1,2,
127             [[C11, J, C13, C14, C15],
128             [C21, C22, C23, C24, C25],
129             [C31, C32, C33, C34, C35],
130             [C41, C42, C43, C44, C45]]):- C22 \= v, C32
                   \= v, C42 \= v.
131
```

```
132 joga_vazio([[C11, C12, C13, C14, C15],
133             [C21, v, C23, C24, C25],
134             [C31, C32, C33, C34, C35],
135             [C41, C42, C43, C44, C45]],
136             J,
137             2,2,
138             [[C11, C12, C13, C14, C15],
139             [C21, J, C23, C24, C25],
140             [C31, C32, C33, C34, C35],
141             [C41, C42, C43, C44, C45]]):- C32 \= v,
                   C42 \= v.
142
143 joga_vazio([[C11, C12, C13, C14, C15],
144             [C21, C22, C23, C24, C25],
145             [C31, v, C33, C34, C35],
146             [C41, C42, C43, C44, C45]],
147             J,
148             3,2,
149             [[C11, C12, C13, C14, C15],
150             [C21, C22, C23, C24, C25],
151             [C31, J, C33, C34, C35],
152             [C41, C42, C43, C44, C45]]):- C42 \= v.
153
154 joga_vazio([[C11, C12, C13, C14, C15],
155             [C21, C22, C23, C24, C25],
156             [C31, C32, C33, C34, C35],
157             [C41, v, C43, C44, C45]],
158             J,
159             4,2,
160             [[C11, C12, C13, C14, C15],
161             [C21, C22, C23, C24, C25],
162             [C31, C32, C33, C34, C35],
163             [C41, J, C43, C44, C45]]).
164
165 joga_vazio([[C11, C12, v, C14, C15],
166             [C21, C22, C23, C24, C25],
167             [C31, C32, C33, C34, C35],
168             [C41, C42, C43, C44, C45]],
169             J,
170             1,3,
171             [[C11, C12, J, C14, C15],
```

```
172                 [C21, C22, C23, C24, C25],
173                 [C31, C32, C33, C34, C35],
174                 [C41, C42, C43, C44, C45]]):- C23 \= v, C33
                        \= v, C43 \= v.
175
176 joga_vazio([[C11, C12, C13, C14, C15],
177                 [C21, C22, v, C24, C25],
178                 [C31, C32, C33, C34, C35],
179                 [C41, C42, C43, C44, C45]],
180                J,
181                2,3,
182                [[C11, C12, C13, C14, C15],
183                [C21, C22, J, C24, C25],
184                [C31, C32, C33, C34, C35],
185                [C41, C42, C43, C44, C45]]):-  C33 \= v,
                      C43 \= v.
186
187 joga_vazio([[C11, C12, C13, C14, C15],
188                 [C21, C22, C23, C24, C25],
189                 [C31,C32, v, C34, C35],
190                 [C41, C42, C43, C44, C45]],
191                J,
192                3,3,
193                [[C11, C12, C13, C14, C15],
194                [C21, C22, C23, C24, C25],
195                [C31, C32, J, C34, C35],
196                [C41, C42, C43, C44, C45]]):-  C43 \= v.
197
198 joga_vazio([[C11, C12, C13, C14, C15],
199                 [C21, C22, C23, C24, C25],
200                 [C31, C32, C33, C34, C35],
201                 [C41, C42, v, C44, C45]],
202                J,
203                4,3,
204                [[C11, C12, C13, C14, C15],
205                [C21, C22, C23, C24, C25],
206                [C31, C32, C33, C34, C35],
207                [C41, C42, J, C44, C45]]).
208
209 joga_vazio([[C11, C12, C13, v, C15],
210                 [C21, C22, C23, C24, C25],
```

```
211                 [C31, C32, C33, C34, C35],
212                 [C41, C42, C43, C44, C45]],
213                J,
214                1,4,
215                [[C11, C12, C13, J, C15],
216                [C21, C22, C23, C24, C25],
217                [C31, C32, C33, C34, C35],
218                [C41, C42, C43, C44, C45]]):- C24 \= v, C34
                       \= v, C44 \= v.
219
220 joga_vazio([[C11, C12, C13, C14, C15],
221                 [C21, C22, C23, v, C25],
222                 [C31, C32, C33, C34, C35],
223                 [C41, C42, C43, C44, C45]],
224                J,
225                2,4,
226                [[C11, C12, C13, C14, C15],
227                [C21, C22, C23, J, C25],
228                [C31, C32, C33, C34, C35],
229                [C41, C42, C43, C44, C45]]):-  C34 \= v,
                      C44 \= v.
230
231 joga_vazio([[C11, C12, C13, C14, C15],
232                 [C21, C22, C23, C24, C25],
233                 [C31, C32, C33, v, C35],
234                 [C41, C42, C43, C44, C45]],
235                J,
236                3,4,
237                [[C11, C12, C13, C14, C15],
238                [C21, C22, C23, C24, C25],
239                [C31, C32, C33, J, C35],
240                [C41, C42, C43, C44, C45]]):-  C44 \= v.
241
242 joga_vazio([[C11, C12, C13, C14, C15],
243                 [C21, C22, C23, C24, C25],
244                 [C31, C32, C33, C34, C35],
245                 [C41, C42, C43, v, C45]],
246                J,
247                4,4,
248                [[C11, C12, C13, C14, C15],
249                [C21, C22, C23, C24, C25],
```

```
250             [C31, C32, C33, C34, C35],
251             [C41, C42, C43, J, C45]]).
252
253 joga_vazio([[C11, C12, C13, C14, v],
254             [C21, C22, C23, C24, C25],
255             [C31, C32, C33, C34, C35],
256             [C41, C42, C43, C44, C45]],
257             J,
258             1,5,
259             [[C11, C12, C13, C14, J],
260             [C21, C22, C23, C24, C25],
261             [C31, C32, C33, C34, C35],
262             [C41, C42, C43, C44, C45]]):- C25 \= v, C35
                    \= v, C45 \= v.
263
264 joga_vazio([[C11, C12, C13, C14, C15],
265             [C21, C22, C23, C24, v],
266             [C31, C32, C33, C34, C35],
267             [C41, C42, C43, C44, C45]],
268             J,
269             2,5,
270             [[C11, C12, C13, C14, C15],
271             [C21, C22, C23, C24, J],
272             [C31, C32, C33, C34, C35],
273             [C41, C42, C43, C44, C45]]):-  C35 \= v,
                  C45 \= v.
274
275 joga_vazio([[C11, C12, C13, C14, C15],
276             [C21, C22, C23, C24, C25],
277             [C31, C32, C33, C34, v],
278             [C41, C42, C43, C44, C45]],
279             J,
280             3,5,
281             [[C11, C12, C13, C14, C15],
282             [C21, C22, C23, C24, C25],
283             [C31, C32, C33, C34, J],
284             [C41, C42, C43, C44, C45]]):-  C45 \= v.
285
286 joga_vazio([[C11, C12, C13, C14, C15],
287             [C21, C22, C23, C24, C25],
288             [C31, C32, C33, C34, C35],
```

```
289                 [C41, C42, C43, C44, v]],
290                 J,
291                 4,5,
292                 [[C11, C12, C13, C14, C15],
293                 [C21, C22, C23, C24, C25],
294                 [C31, C32, C33, C34, C35],
295                 [C41, C42, C43, C44, J]]).
```

2.2 Anexo 2 - minimax.pl

```
1  joga :- estado_inicial(Ei), joga(Ei).
2
3  pede_coluna(Coluna):-
4      read(Coluna).
5
6  joga_pc(Ei):-terminal(Ei),write("Game Over!!!!").
7  joga_pc(Ei):-minimax_decidir(Ei, joga(X,Y)),
8                write(joga(X,Y)),nl,
9                joga_vazio(Ei, x, X, Y, Enn),
10               joga(Enn).
11
12 joga(Ei):- terminal(Ei),printTable(Ei),write("Game Over
       !!!!").
13 joga(Ei) :-printTable(Ei),
14              pede_coluna(Coluna),
15              joga_vazio(Ei, o, _, Coluna, En),
16              printTable(En),
17              joga_pc(En).
18
19 printTable([]).
20 printTable([T|Tl]):-
21     write(T),nl,
22     printTable(Tl).
23
24 % decide qual     a melhor jogada num estado do jogo
25 % minimax_decidir(Estado, MelhorJogada)
26
27 % se     estado terminal n o h   jogada
28 minimax_decidir(Ei,terminou) :- terminal(Ei).
29
30 % Para cada estado sucessor de Ei calcula o valor
       minimax do estado
31 % Opf     o operador (jogada) que tem maior valor
32 % Nota: assume que o jogador     o "x"
33 minimax_decidir(Ei,Opf) :-
34          findall(Vc-Op, (oper(Ei,x,Op,Es), minimax_valor
                (Es,Vc,1)), L),
35          escolhe_max(L,Opf).
36
```

```
37 % se um estado     terminal o valor     dado pela
       fun   o de utilidade
38 % Nota: assume que o jogador     o "x"
39 minimax_valor(Ei,Val,_) :-
40         terminal(Ei),
41         valor(Ei,Val).
42
43 %Se o estado n o     terminal o valor    :
44 % -se a profundidade     par, o maior valor dos
       sucessores de Ei
45 % -se aprofundidade     impar o menor valor dos
       sucessores de Ei
46 minimax_valor(Ei,Val,P) :-
47         P1 is P+1, jogador(P1,J),
48         findall(Val1, (oper(Ei,J,_,Es), minimax_valor(
             Es,Val1,P1)), V),
49         seleciona_valor(V,P,Val).
50
51
52 % jogador "x" nas jogadas impares e jogador "o" nas
       jogadas pares
53 jogador(P, o) :- X is P mod 2, X = 0.
54 jogador(P, x) :- X is P mod 2, X = 1.
55
56 % Se a profundidade (P)     par, retorna em Val o maximo
        de V
57 seleciona_valor(V,P,Val) :-
58         X is P mod 2, X=0,!,
59         maximo(V,Val).
60
61 % Sen o retorna em Val o minimo de V
62 seleciona_valor(V,_,Val):- minimo(V,Val).
63
64 %% Predicados auxiliares
65
66 escolhe_max([A|R],Val):- escolhe_max(R,A,Val).
67
68 escolhe_max([],_-Op,Op).
69 escolhe_max([A-_|R],X-Op,Val) :- A < X,!, escolhe_max(R
       ,X-Op,Val).
70 escolhe_max([A|R],_,Val):- escolhe_max(R,A,Val).
```

```
71
72
73 maximo([A|R],Val):- maximo(R,A,Val).
74
75 maximo([],A,A).
76 maximo([A|R],X,Val):- A < X,!, maximo(R,X,Val).
77 maximo([A|R],_,Val):- maximo(R,A,Val).
78
79 minimo([A|R],Val):- minimo(R,A,Val).
80
81 minimo([],A,A).
82 minimo([A|R],X,Val):- A > X,!, minimo(R,X,Val).
83 minimo([A|R],_,Val):- minimo(R,A,Val).
```

2.3 Anexo 3 - alfabeta.pl

```
1  joga :- estado_inicial(Ei), joga(Ei).
2
3  pede_coluna(Coluna):-
4      read(Coluna).
5
6  joga_pc(Ei):-terminal(Ei),write("Game Over!!!!").
7  joga_pc(Ei):-alfabeta(Ei, joga(X,Y)),
8                write(joga(X,Y)),nl,
9                joga_vazio(Ei, x, X, Y, Enn),
10               joga(Enn).
11
12 joga(Ei):- terminal(Ei),printTable(Ei),write("Game Over
       !!!!").
13 joga(Ei) :-printTable(Ei),
14              pede_coluna(Coluna),
15              joga_vazio(Ei, o, _, Coluna, En),
16              printTable(En),
17              joga_pc(En).
18
19 printTable([]).
20 printTable([T|Tl]):-
21     write(T),nl,
22     printTable(Tl).
23
24 % decide qual     a melhor jogada num estado do jogo
25 % alfabeta(Estado, MelhorJogada)
26
27 % se     estado terminal n o h   jogada
28 alfabeta(Ei,terminou) :- terminal(Ei).
29
30 % Nota: assume que o jogador     o "x"
31 alfabeta(Ei,Opf) :-
32          findall(Vc-Op, (oper(Ei,x,Op,Es), alfabeta_min(
               Es,Vc,1,-10000,10000)), L),
33          escolhe_max(L,Opf).
34
35 % se um estado     terminal o valor     dado pela
       fun   o de utilidade
36 % Nota: assume que o jogador     o "x"
```

```
37 alfabeta_min(Ei,Val,_,_,_) :-
38         terminal(Ei),
39         valor(Ei,Val), !.
40
41 alfabeta_min(Ei,Val,P,Alfa,Beta) :-
42         P1 is P+1, jogador(P1,J),
43         V is 10000,
44         findall(Es, oper(Ei,J,_,Es), L),
45         processa_lista_min(L, P1, V, Alfa, Beta, Val),
               !.
46
47 processa_lista_min([], _, V, _, _, V).
48 processa_lista_min([H|T], P, V, A, B, V1) :-
49         alfabeta_max(H, V2, P, -10000, 10000),
50         min(V, V2, V3),
51         (V3 < A, V1 is V3; min(B, V3, B1),
              processa_lista_min(T, P, V3, A, B1, V1)).
52
53 min(A,B,A) :- A < B, !.
54 min(_, B, B).
55
56 alfabeta_max(Ei,Val,_,_,_) :-
57         terminal(Ei),
58         valor(Ei,Val), !.
59
60 alfabeta_max(Ei,Val,P,Alfa,Beta) :-
61         P1 is P+1, jogador(P1,J),
62         V is -10000,
63         findall(Es, oper(Ei,J,_,Es), L),
64         processa_lista_max(L, P1, V, Alfa, Beta, Val),
               !.
65
66 processa_lista_max([], _, V, _, _, V).
67 processa_lista_max([H|T], P, V, A, B, V1) :-
68         alfabeta_min(H, V2, P, -10000, 10000),
69         max(V, V2, V3),
70         (V3 >= B, V1 is V3; max(A, V3, A1),
              processa_lista_max(T, P, V3, A1, B, V1)).
71
72 max(A,B,B) :- A < B, !.
73 max(A, _, A).
```

```
74
75 % jogador "x" nas jogadas impares e jogador "o" nas
       jogadas pares
76 jogador(P, o) :- X is P mod 2, X = 0.
77 jogador(P, x) :- X is P mod 2, X = 1.
78
79 % Se a profundidade (P)     par, retorna em Val o maximo
       de V
80 seleciona_valor(V,P,Val) :-
81         X is P mod 2, X=0,!,
82         maximo(V,Val).
83
84 % Sen o retorna em Val o minimo de V
85 seleciona_valor(V,_,Val):- minimo(V,Val).
86
87 %% Predicados auxiliares
88
89 escolhe_max([A|R],Val):- escolhe_max(R,A,Val).
90
91 escolhe_max([],_-Op,Op).
92 escolhe_max([A-_|R],X-Op,Val) :- A < X,!, escolhe_max(R
       ,X-Op,Val).
93 escolhe_max([A|R],_,Val):- escolhe_max(R,A,Val).
94
95
96 maximo([A|R],Val):- maximo(R,A,Val).
97
98 maximo([],A,A).
99 maximo([A|R],X,Val):- A < X,!, maximo(R,X,Val).
100 maximo([A|R],_,Val):- maximo(R,A,Val).
101
102 minimo([A|R],Val):- minimo(R,A,Val).
103
104 minimo([],A,A).
105 minimo([A|R],X,Val):- A > X,!, minimo(R,X,Val).
106 minimo([A|R],_,Val):- minimo(R,A,Val).
```