

Relatório Intercalar de PL

11/10/2002

Elementos de identificação do trabalho e do grupo

Nelson Jorge Silva Rodrigues – ei00070@fe.up.pt
Ricardo Jorge Marques Veloso – ei00125@fe.up.pt

O trabalho consiste na realização de um jogo de tabuleiro para dois jogadores. O módulo de jogo deverá ser implementado em Prolog devendo ser complementado com um visualizador em Java3D, contendo também este módulo a interface com o utilizador.

O jogo escolhido por este grupo foi o jogo de damas.

Resumo

O jogo, muito sumariamente, resume-se a um tabuleiro em xadrez de 8x8 casas numeradas alfabeticamente nas linhas e com números nas colunas, as casas serão alternadamente brancas ou pretas, sendo as peças (“pedras”) colocadas nas casas pretas. Cada jogador inicia o jogo com um conjunto de 12 pedras de uma determinada cor, brancas ou pretas. O objectivo do jogo é impedir que o jogador adversário tenha jogadas possíveis a efectuar, sendo a maneira mais simples de o conseguir “comer” todas as peças do adversário. Caso nenhum jogador atinja o objectivo o jogo é considerado empatado.

Existem dois tipos de jogadas possíveis: jogada de movimentação, em que um jogador movimenta a pedra de uma casa para outra casa que esteja na diagonal da primeira, sempre na direcção em que o jogador esteja voltado; jogada de comer em que o jogador cumpre a regra da diagonal e da direcção, mas que pode avançar sobre uma peça da cor oposta, avançando assim duas casas, este tipo de jogada pode ser encadeada, sendo possível efectuar várias jogadas de comer numa única movimentação.

No caso de uma peça de um determinado jogador atinja a última linha do lado oposto em que iniciou o jogo, essa peça recebe a denominação de “dama”, e representa-se pela sobreposição de duas pedras normais. A dama além de poder efectuar todas as jogadas de uma pedra normal, pode ainda movimentar-se mais que uma casa de cada vez, e no sentido inverso à movimentação normal, tendo no entanto de cumprir a regra da diagonal.

Bibliografia

- Leon Sterling, Ehud Shapiro. The Art of Prolog, 2a edição. MIT Press, 1994.
- Adventure In Prolog by AMZI –

<http://oopweb.com/Prolog/Documents/AdventureInProlog/VolumeFrames.html>

Perspectivas de Desenvolvimento

Após termos efectuado uma cuidada análise do jogo que nos propusemos a desenvolver, chegamos a conclusão que este poderá não ser o jogo simples que à partida somos levados a pensar, a quantidade e diversidade das jogadas possíveis podem tornar este jogo bastante difícil de implementar.

O jogo vai ser desenvolvido tendo em conta as regras já enunciadas, e tentando desenvolver uma componente de inteligência artificial minimamente desafiante para o jogador, mas sem nunca esquecer que o nosso objectivo é atingir um bom nível na qualidade dos predicados desenvolvidos e nas estruturas de dados aplicadas.

Neste momento estimamos que estamos algures nos 20% de trabalho concluído, visto que já implementamos todos os predicados de alteração da estrutura que representa o tabuleiro, e já implementamos algumas das regras necessárias para validar uma jogada.

Código já desenvolvido

```
1  %'@' - black
2  %'O' - white
3  %' ' - espaco
4  %'B' - Dama preta
5  %'W' - Dama branca
6
7  tabuleiro([
8      ['@','@','@','@','@','@','@','@'],
9      ['@','@','@','@','@','@','@','@'],
10     [' ','@',' ','@',' ','@',' ','@'],
11     [' ',' ','@',' ',' ','@',' ',' '],
12     [' ',' ',' ','@',' ',' ','@',' '],
13     [' ',' ',' ',' ','@',' ',' ','@'],
14     [' ',' ',' ',' ','@',' ',' ','@'],
15     [' ',' ',' ',' ','@',' ',' ','@']
16     ]).
17
18  /*****
19  /*      Funcoes de entrada de dados      */
20  /*****
21
22  inicio :-
23      write('1- Humano/Humano'),nl,
24      write('2- Humano/Computador'),nl,
25      write('3- Computador/Humano'),nl,
26      write('4- Computador/Computador'),nl,
27      read(Opcao),
28      tabuleiro(Board),
29      joga('brancas', Board).
30
31
32  /*****
33  /*      Funcao do tipo de jogo      */
34  /*****
35
36  joga(Jogador, Board) :-
37      showBoard(Board),
38      recebeJogada(Jogador, Board, NewBoard ),
39      mudaJogador(Jogador, Jogador2),joga(Jogador2, NewBoard).
40
41  mudaJogador('brancas', 'pretas').
42  mudaJogador('pretas', 'brancas').
43
44  /*****
45  /*      Funcoes do tabuleiro      */
46  /*****
47
48  showBoard(Board) :-
49      nl,
50      showLines( Board, 1 ),
51      write(' A B C D E F G H'),nl.
52
53
```

```

54 showLines( [], _ ).
55 showLines( [Cabeca | Resto], N ) :-
56     write(N),
57     write(' '),
58     N1 is N + 1,
59     showLine(Cabeca),nl,
60     showLines(Resto, N1).
61
62
63 showLine( [] ).
64 showLine( [Cabeca | Resto] ) :-
65     %   passa( X, Cabeca ),
66     write(Cabeca),
67     write(' '),
68     showLine(Resto).
69
70 /*****
71 /*           Funcoes de jogada           */
72 /*****
73
74 recebeJogada(Jogador, Board, NewBoard) :-
75     repeat,
76     write('Jogada '),write(Jogador),write(' - '),
77     read(X),name(X,Lista),modificaLista( Lista, ListaFinal),
78     validaJogada( Jogador, ListaFinal, Board, NewBoard ).
79
80
81
82
83 /*****
84 /*           Funcao para validar jogada           */
85 /*****
86
87 validaJogada( Jogador, [ColIni, LinIni, ColFim, LinFim], Board, NewBoard
) :-!, (
88     verificaLinha( Jogador, LinIni, LinFim ),
89     verificaColuna( ColIni, ColFim ),
90     descobrePeca( Jogador, Peca ),
91     membroBoard( Peca, ColIni, LinIni, Board ),
92     membroBoard( '_', ColFim, LinFim, Board ),
93     fazJogada( Jogador, 'simples', ColIni, LinIni, ColFim, LinFim,
Board, NewBoard )
94     ;
95     determinaIntermedia( Jogador, ColIni, LinIni, ColFim, LinFim, X, Y
),
96     verificaLinha( Jogador, LinIni, Y ), verificaLinha( Jogador, Y,
LinFim ),
97     verificaColuna( ColIni, X ), verificaColuna( X, ColFim ),
98     descobrePeca( Jogador, Peca ),
99     membroBoard( Peca, ColIni, LinIni, Board ),
100    mudaJogador( Jogador, Adversario), descobrePeca( Adversario,
PecaAdv ),
101    membroBoard( PecaAdv, X, Y, Board ),
102    membroBoard( '_', ColFim, LinFim, Board ),
103    fazJogada( Jogador, 'come', ColIni, LinIni, X, Y, ColFim, LinFim,
Board, NewBoard )
104    ;

```

```

105         write( 'Jogada Invalida' ), nl, fail).
106
107 validaJogada( Jogador, [ColIni, LinIni, ColFim, LinFim | Resto ], Board,
NewBoard ) :-
108     validaJogada( Jogador, [ColIni, LinIni, ColFim, LinFim], Board,
NewBoard1 ),
109     validaJogada( Jogador, [ColFim, LinFim | Resto ], NewBoard1,
NewBoard ).
110
111
112 descobrePeca( 'brancas', '0' ).
113 descobrePeca( 'pretas', '@' ).
114
115
116 membroBoard(Cor, Col, Lin, Board) :-
117     linhaBoard(Linha, Lin, 1, Board),
118     linhaBoard(Cor, Col, 1, Linha).
119
120 linhaBoard(Member, N, N, [Member | R]).
121 linhaBoard(Member, N, P, [Cabeca | R]) :-
122     P1 is P + 1,
123     linhaBoard(Member, N, P1, R).
124
125 verificaLinha('brancas', LinIni, LinFim ) :-
126     X is LinIni - 1,
127     X == LinFim.
128 verificaLinha( 'pretas', LinIni, LinFim ) :-
129     X is LinIni + 1,
130     X == LinFim.
131
132 verificaColuna( 1, 2 ).
133 verificaColuna( 8, 7 ).
134 verificaColuna( ColIni, ColFim ) :-
135     Z is ColIni + 1, ColFim == Z
136     ;
137     Z is ColIni - 1, ColFim == Z .
138
139 determinaIntermedia( 'brancas', ColIni, LinIni, ColFim, LinFim, X, Y ) :-
140     (ColFim > ColIni, X is ColFim - 1 ; X is ColIni - 1),
141     (Y is LinIni - 1 ).
142
143 determinaIntermedia( 'pretas', ColIni, LinIni, ColFim, LinFim, X, Y ) :-
144     (ColFim > ColIni, X is ColFim - 1 ; X is ColIni - 1),
145     (Y is LinIni + 1 ).
146
147 /*****
148 /*          Funcao para efectuar jogada          */
149 /*****
150
151
152 fazJogada( Jogador, 'simples', ColIni, LinIni, ColFim, LinFim, Board,
NewBoard ) :-
153     descobrePeca( Jogador, Peca ),
154     mudaBoard( '_', ColIni, LinIni, Board, NewBoard1 ),
155     mudaBoard( Peca, ColFim, LinFim, NewBoard1, NewBoard ).
156

```

```

157 fazJogada( Jogador, 'come', ColIni, LinIni,X, Y, ColFim, LinFim, Board,
NewBoard ) :-
158     fazJogada( Jogador, 'simples', ColIni, LinIni, X, Y, Board,
NewBoard1 ),
159     fazJogada( Jogador, 'simples', X, Y, ColFim, LinFim, NewBoard1,
NewBoard ).
160
161 mudaBoard( Novo, Col, Lin, Board, NewBoard ) :-
162     mudaBoard2( 1, Novo, Col, Lin, Board, NewBoard ).
163
164 mudaBoard2( _,_,_,_, [],[] ).
165 mudaBoard2(Y, Novo, X, Y, [Lin|Resto], [NovLin|NovResto]) :-
166     mudaLinha( 1, Novo, X, Lin, NovLin ),
167     N2 is Y + 1,
168     mudaBoard2( N2, Novo, X, Y, Resto, NovResto ).
169 mudaBoard2(N, Novo, X, Y, [Lin|Resto], [Lin|NovResto] ) :-
170     Y\=N, N2 is N + 1,
171     mudaBoard2( N2, Novo, X, Y, Resto, NovResto ).
172
173 mudaLinha( _,_,_, [], [] ).
174 mudaLinha( X, Novo, X, [Peca|Resto], [Novo|NovResto] ) :-
175     N2 = X + 1,
176     mudaLinha( N2, Novo, X, Resto, NovResto ).
177 mudaLinha( N, Novo, X, [Peca|Resto], [Peca|NovResto] ) :-
178     N \= X, N2 is N + 1,
179     mudaLinha( N2, Novo, X, Resto, NovResto ).
180
181
182 /*****
183 /*      Funcao para traduzir asciis em numeros      */
184 /*****
185
186 modificaLista( [], [] ).
187 modificaLista( [Col, Lin |Resto], [NovaCol, NovaLinha|NovoResto] ) :-
188     modifica( Col, Lin, NovaCol, NovaLinha ),
189     modificaLista( Resto, NovoResto ).
190
191 modifica( X, Y, Col, Lin ) :-
192     Col is X - 96,
193     Lin is Y - 48.

```