**Distrify**

Relatório Intercalar



Mestrado Integrado em Engenharia Informática e Computação

Programação em Lógica

**Grupo Distrify\_2:**

André Rodrigues Barros – up201303567

Edgar Duarte Ramos – up201305973

Faculdade de Engenharia da Universidade do Porto

Rua Roberto Frias, sn, 4200-465 Porto, Portugal

11 de Setembro de 2015

**Índice**

1. O jogo Distrify 3
2. Representação do estado do jogo 4 a 6
3. Visualização do Tabuleiro 7 a 9
4. Jogadas 10

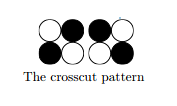
**1 - O jogo Distrify**

Distrify é um jogo de conexão jogado por dois jogadores, constituído por peças pretas e brancas, num tabuleiro de 9x9 até 19x19.

O jogador que controla as peças pretas joga primeiro, colocando uma peça preta em qualquer local do tabuleiro. As jogadas seguintes são alternadas entre os jogadores e cada jogada consiste na colocação de uma ou duas peças em qualquer espaço livre. No caso em que sejam jogadas duas peças:

a)só podem ser jogadas na vertical ou horizontal

b) não podem resultar na formação de um *triplet* ou *crosscut.*

Um *triplet* refere-se a três peças da mesma cor seguidas em qualquer direção.

Um *crosscut* refere-se a um bloco de 2x2 em que cantos opostos são controlados por jogadores diferentes

O jogador que controla as peças pretas ganha se, em qualquer momento do jogo, existir um caminho de peças pretas que se prolongue desde a linha superior até à inferior. As peças brancas ganham se houver um caminho que ligue a coluna mais à esquerda até à mais à direita.

**2 - Representação do Estado de jogo**

Representação do estado inicial do jogo:

[ [0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0],

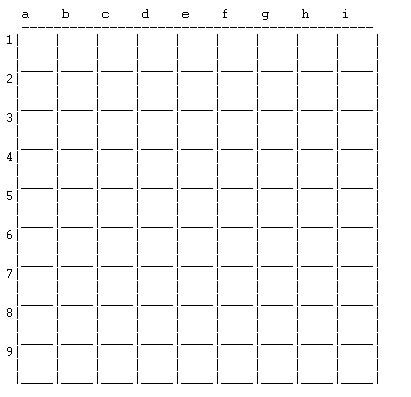
[0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0],]



Representação de um estado intermédio do jogo:

[ [1,0,0,0,0,0,0,0,0],

[0,2,2,0,0,0,0,0,0],

[2,0,0,0,0,0,0,0,0],

[0,2,0,0,1,0,0,0,0],

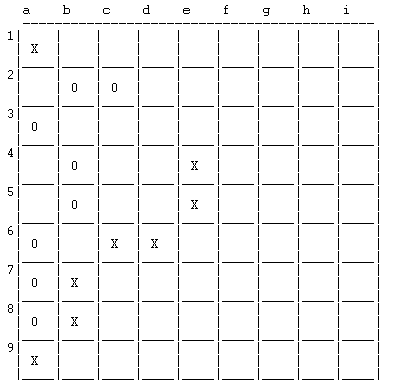
[0,2,0,0,1,0,0,0,0],

[2,0,1,1,0,0,0,0,0],

[2,1,0,0,0,0,0,0,0],

[2,1,0,0,0,0,0,0,0],

[1,0,0,0,0,0,0,0,0]]



Representação de um estado final do jogo:

[ [1,0,0,2,2,1,0,0,0],

[0,2,2,0,0,1,0,0,0],

[2,0,0,0,0,1,0,0,0],

[0,2,0,0,1,0,0,0,0],

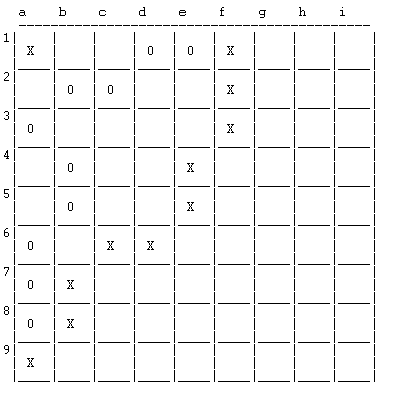
[0,2,0,0,1,0,0,0,0],

[2,0,1,1,0,0,0,0,0],

[2,1,0,0,0,0,0,0,0],

[2,1,0,0,0,0,0,0,0],

[1,0,0,0,0,0,0,0,0] ]



**3 - Visualização do Tabuleiro**

Para mostrar o tabuleiro de jogo, será utilizado, em princípio, o seguinte código:

starting\_state([ [0,0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0,0] ]).

Num estado inicial define o estado inicial do tabuleiro.

printColumnId:- write(' a b c d e f g h i \n').

printInitialSeperator:- write( ' -------------------------------------------- \n').

Posteriormente, preenchemos a linha de cabeçalho do tabuleiro com as letras que servirão de guia para a colocação das peças no mesmo, aquando do começo do jogo. Após essa linha imprime-se uma linha continua para começar as divisórias do tabuleiro.

Três predicados onde se define o que apresentar no tabuleiro quando colocamos os diferentes números apresentados (0, 1 ou 2)

write\_piece(0):-write('| ').

write\_piece(1):-write('| X ').

write\_piece(2):-write('| 0 ').

write\_line([]):- write('|').

write\_line([H|T]):-

write\_piece(H),

write\_line(T).

Esta secção do código apresentado, por sua vez, imprime uma linha do tabuleiro.

show\_lines(\_, []).

show\_lines(N, [H|T]):-

write(N), write('| | | | | | | | | | \n'),write(' '), write\_line(H), nl, write('|\_\_\_\_|\_\_\_\_|\_\_\_\_|\_\_\_\_|\_\_\_\_|\_\_\_\_|\_\_\_\_|\_\_\_\_|\_\_\_\_|', nl,

N2 is N+1,

show\_lines(N2, T).

Usando os predicados apresentados anteriormente (write\_line), o predicado show\_lines imprime todas as linhas do tabuleiro, bem como as divisórias necessárias para que a apresentação do tabuleiro seja a mais correta.

Secção do código encarregue de imprimir o tabuleiro completo. De notar que o predicado show\_board usa o show\_lines.

show\_board(Board):-

nl, printColumnId, printInitialSeperator,

show\_lines(1, Board), nl.

start:- starting\_state(Board), show\_board(Board).

**4 - Jogadas**

Cabeçalho do predicado de colocação de uma peça:

One\_piece(Row, Column, Board)

Cabeçalho do predicado de colocação de duas peças:

Two\_pieces(Row, Column, Row, Column, Board)