T´ıtulo do Trabalho

Relat´orio Intercalar



Mestrado Integrado em Engenharia Informática e

Computa¸cão

Programa¸cao em Lógica

Grupo Pentalath 2:

Daniela Quintas Fernandes de Sá - 201405457

Tiago Rafael Ferreira da Silva - 201402841

Faculdade de Engenharia da Universidade do Porto

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

14 de Outrubro de 2016

1. O Jogo Pentalath

História

Pentalath foi criado por um programa chamado Ludi, que por sua vez foi desenvolvido por Cameron Browne, em Dezembro de 2007.

“Ludi” cria conjuntos de regras usando combinações de regras de jogos já existentes. Depois determina se a combinação obtida é viável e interessante para ser jogada por humanos.

Enquanto que o Pentalath era inicialmente jogado num tabuleiro trapezoidal, este foi trocado por um hexagonal pois são mais comuns.

Regras

O objetivo do jogo é ser o primeiro a formar uma linha reta de cinco ou mais peças de uma cor no seguinte tabuleiro:

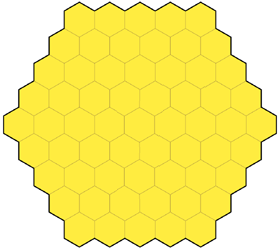


Fig. 1 – O tabuleiro vazio

Os jogadores colocam peças no tabuleiro à vez, sendo que inicialmente o tabuleiro está vazio. Imediatamente após a jogada de abertura o segundo jogador tem a possibilidade de trocar de cor com o primeiro. Isto serve para incentivar uma primeira jogada comedida.

É possível capturar peças do adversário. Se uma série de peças for rodeada por peças adversárias, elas são retiradas do tabuleiro, como ilustrado na figura seguinte:

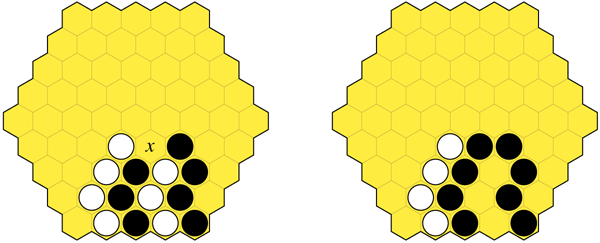


Fig. 2 – Situação de captura em Pentalath

Não é permitido ao jogador colocar a peça numa posição sem liberdade. Isto é, já rodeada de peças inimigas.

Fonte: <http://www.cameronius.com/games/pentalath/>

1. Representação do Estado de Jogo

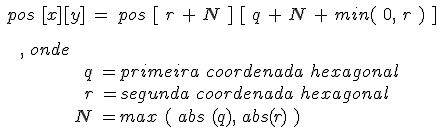
Para representar o nosso tabuleiro hexagonal usamos uma lista de listas. Optamos por usar listas de tamanho igual à maior linha do tabuleiro e mapeamo-las usando a expressão:

Fig. 3 – Conversão das coordenadas hexagonais para axiais

Temos então um array bidimensional vazio de início:

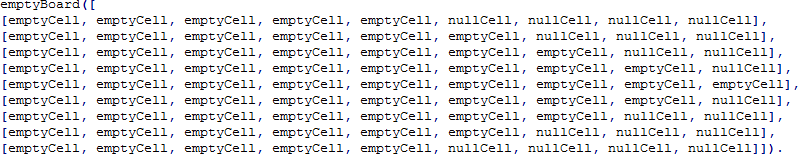
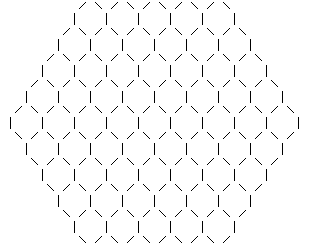
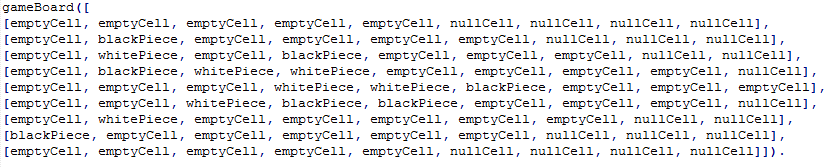


Fig. 4 (cima) – Código Prolog do array vazio utilizado

Fig. 5 (esquerda) – Representação gráfica resultante do tabuleiro vazio

Ao longo do jogo vão sendo adicionadas peças, que representamos na matriz por “blackPiece” e “whitePiece”, como ilustrado a seguir:



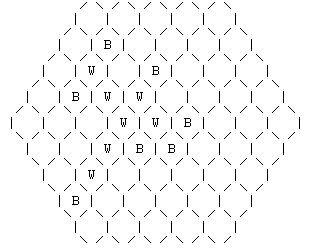
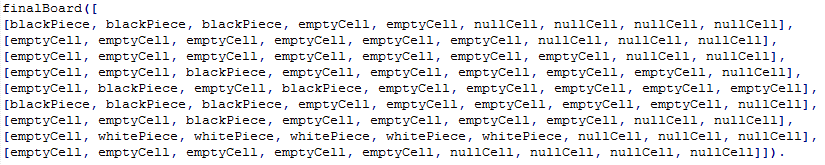


Fig. 6 (cima) – Código Prolog de um possível array do tabuleiro em jogo

Fig. 7 (esquerda) – Representação gráfica resultante do código na figura 6

E finalmente temos um estado final em que um dos jogadores ganhou, por exemplo:



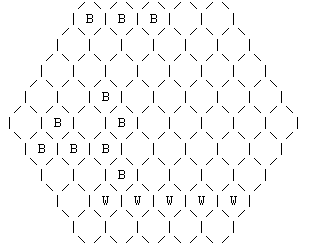


Fig. 8 (cima) – Código Prolog de um possível array do tabuleiro final

Fig. 9 (esquerda) – Representação gráfica resultante do código na figura 8

1. Visualização do Tabuleiro

O tabuleiro terá, então, a seguinte representação gráfica:

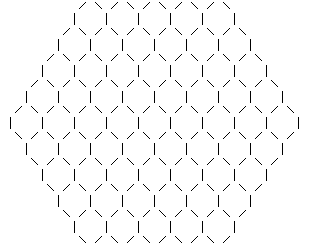
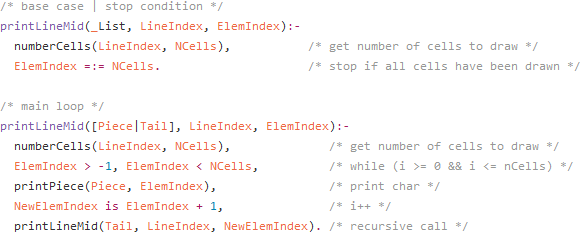
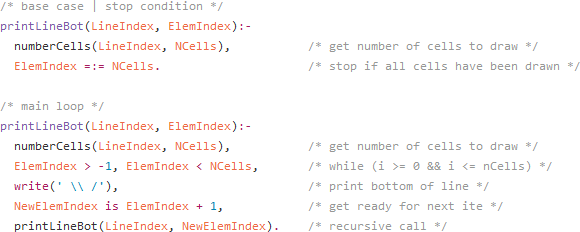
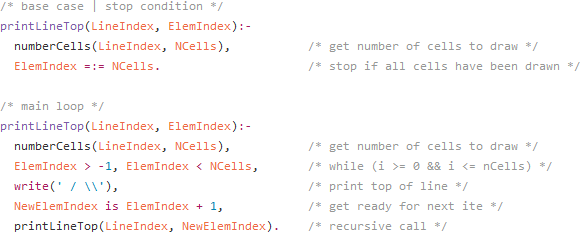


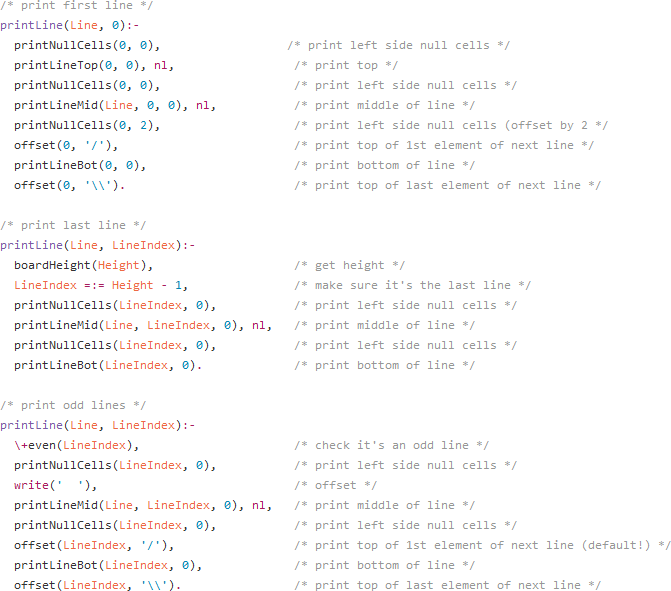
Fig. 10 (esquerda) – Representação gráfica do tabuleiro

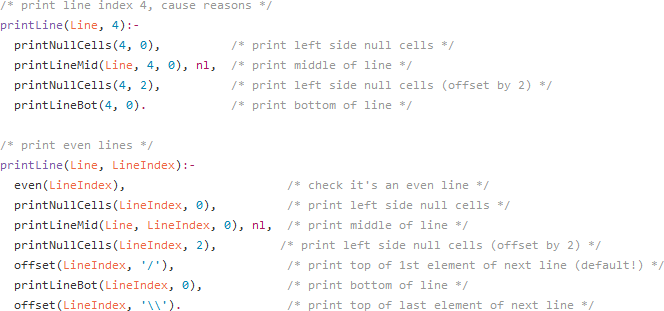
E para tal representação utilizamos o seguinte código em Prolog:











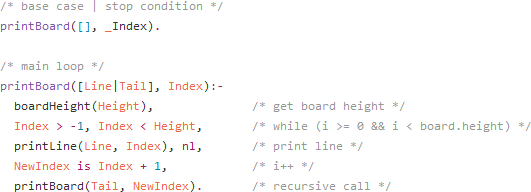


Fig. 11 – Código Prolog para impressão do tabuleiro

O código funciona com base no output de espaços e “frações de hexágono”. Como é possível observar na figura 11, imprimimos de forma diferente as linhas ímpares, pares, a primeira e a última.

1. Movimentos

É possível colocar uma peça por jogada em qualquer casa do tabuleiro, exceto se esta já estiver ocupada ou rodeada de peças inimigas.

Para tal, vamos usar o predicado Prolog com o seguinte cabeçalho:

placePiece(Player, Q, R) ,

onde Player indica a cor da peça a ser colocada e Q e R são as coordenadas hexagonais da posição escolhida.