Oolong

Relatório Intercalar

Mestrado Integrado em Engenharia Informática e Computação

Programação em Lógica

Grupo Oolong_3:
Ricardo Miguel Oliveira Rodrigues de Carvalho - up201503717
Tiago Lascasas dos Santos - up201503616

Faculdade de Engenharia da Universidade do Porto
Rua Roberto Frias, sn, 4200-465 Porto, Portugal

15 de Outubro de 2017

1 O Jogo Oolong

Oolong é um jogo de tabuleiro para 2 jogadores baseado num paradigma de controlo de área, e foi criado em 2017 pelo estúdio independente Black Straw Games. O tema do jogo é uma competição dentro de uma casa de chá em que o objetivo de cada jogador é, jogando à vez, servir o seu chá (verde ou preto) à maioria das pessoas na maioria das mesas.

O tabuleiro é composto por 9 mesas com 9 lugares, dispostas como indicado na figura 1.

Figura 1: Tabuleiro de Oolong.

Cada mesa, exceto a central, tem um marcador especial, que serão descritos posteriormente. As peças são compostas por 40 de cada cor (preto e verde) mais uma peça vermelha, o empregado. O empregado marca qual a mesa em que a próxima jogada será feita e em que lugar o jogador não pode colocar a sua peça.

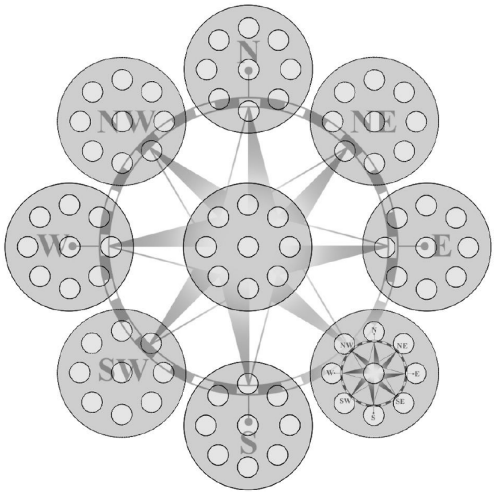

Figura 2: Definição do tabuleiro com pontos cardeais.

O jogo é começado pelo jogador com as peças pretas e com o empregado na posição central da mesa central. Assim, o primeiro jogador tem de jogar numa das 8 posições livres da mesa central e a posição escolhida indicará onde a primeira peça verde será colocada. As mesas e as suas posições estão organizadas segundo os pontos cardeais, com um ponto adicional para sinalizar o centro (Figura 2).

Assim sendo, a jogada seguinte é sempre feita na mesa correspondente ao ponto cardeal onde foi feita a jogada anterior e o empregado é colocado no ponto correspondente à mesa da jogada anterior. Um exemplo do fluxo do jogo pode ser o seguinte:

1. Empregado na mesa Central e casa Central. Jogador preto coloca a peça na casa NW da mesa Central;
2. Empregado na mesa NW e na casa Central. Jogador verde coloca a peça na casa E da mesa NW;
3. Empregado na mesa E e na casa NW. Jogador preto coloca a peça na casa S da mesa E;
4. ...e assim sucessivamente.

O jogo é continuado, à vez, até que um jogador conquiste 5 das 9 mesas. Para conquistar uma mesa, um jogador tem de ocupar pelo menos 5 posições nessa mesa.

Algumas jogadas podem ativar ações especiais, dadas por marcadores que são atribuídos aleatoriamente às mesas, no máximo de um por mesa. As ações possíveis são:

- Um jogador pode mover uma das suas peças de uma mesa para outra sendo que nenhuma destas foi conquistadas.
  - Existem dois marcadores para esta ação, cada um referente a uma das cores das peças dos jogadores.
  - Esta ação é desencadeada com 3 peças da mesma cor na mesa correspondente.
- Um jogador pode mover o empregado da mesa atual para a mesma posição numa mesa à escolha, mudando a mesa em que a próxima jogada é feita.
  - Existem dois marcadores para esta ação, cada um referente a uma das cores das peças dos jogadores.
  - Esta ação é desencadeada com 5 peças da mesma cor na mesa correspondente.
- O jogador que ativa esta ação pode rodar a mesa em qualquer direção (rodando o empregado com ela).
  - Existem dois marcadores para esta ação.
  - Esta ação é desencadeada com 4 peças da mesma cor na mesa correspondente.
- O jogador que ativa esta ação pode trocar quaisquer duas mesas não conquistadas (sem as rodar), movendo também o empregado caso este esteja numa das mesas.

  - Esta ação é desencadeada com 4 peças da mesma cor na mesa correspondente.

- O jogador que ativa esta ação pode trocar qualquer mesa conquistada com uma não conquistada (mantendo a orientação destas), movendo o empregado caso este esteja em qualquer das mesas afetadas.

  - Esta ação é desencadeada com 5 peças da mesma cor na mesa correspondente.

No caso particular de uma jogada (incluindo ações especiais) implicar que o jogador coloque a sua peça numa mesa completa, o jogador pode escolher colocar a peça em qualquer lugar vazio em qualquer das mesas.

2 Representação do Estado do Jogo

Para representar o estado do jogo recorremos à base de dados interna do Prolog, criando um predicado para cada posição: pos(Table, Position, Content). Table indica a mesa, Position a posição dentro da mesa e Content o conteúdo dessa posição. O estado inicial contém todas as posições vazias, o que é internamente representado por um 0, exceto a posição c da mesa c, que contém o empregado (representado por w). Table e Position tomam valores referentes a pontos cardeais (n, s, e, w, nw, ne, sw, se), mais o valor c para representar o centro, seguindo exatamente a mesma distribuição da figura 2.

```
pos(nw, nw, 0). pos(nw, n, 0). pos(nw, ne, 0).      pos( n, nw, 0). pos( n, n, 0). pos( n, ne, 0).      pos(ne, nw, 0). pos(ne, n, 0). pos(ne, ne, 0).
pos(nw,  w, 0). pos(nw, c, 0). pos(nw, e,  0).      pos( n,  w, 0). pos( n, c, 0). pos( n,  e, 0).      pos(ne,  w, 0). pos(ne, c, 0). pos(ne,  e, 0).
pos(nw, sw, 0). pos(nw, s, 0). pos(nw, se, 0).      pos( n, sw, 0). pos( n, s, 0). pos( n, se, 0).      pos(ne, sw, 0). pos(ne, s, 0). pos(ne, se, 0).

pos( w, nw, 0). pos( w, n, 0). pos( w, ne, 0).      pos( c, nw, 0). pos( c, n, 0). pos( c, ne, 0).      pos( e, nw, 0). pos( e, n, 0). pos( e, ne, 0).
pos( w,  w, 0). pos( w, c, 0). pos( w, e,  0).      pos( c,  w, 0). pos( c, c, w). pos( c,  e, 0).      pos( e, w, 0).  pos( e, c, 0). pos( e,  e, 0).
pos( w, sw, 0). pos( w, s, 0). pos( w, se, 0).      pos( c, sw, 0). pos( c, s, 0). pos( c, se, 0).      pos( e, sw, 0). pos( e, s, 0). pos( e, se, 0).

pos(sw, nw, 0). pos(sw, n, 0). pos(sw, ne, 0).      pos( s, nw, 0). pos( s, n, 0). pos( s, ne, 0).      pos(se, nw, 0). pos(se, n, 0). pos(se, ne, 0).
pos(sw,  w, 0). pos(sw, c, 0). pos(sw, e,  0).      pos( s,  w, 0). pos( s, c, 0). pos( s,  e, 0).      pos(se, w, 0).  pos(se, c, 0). pos(se,  e, 0).
pos(sw, sw, 0). pos(sw, s, 0). pos(sw, se, 0).      pos( s, sw, 0). pos( s, s, 0). pos( s, se, 0).      pos(se, sw, 0). pos(se, s, 0). pos(se, se, 0).
```

Figura 3: Estado inicial.

No decorrer do jogo, os predicados vão sendo removidos e substituídos por uns novos predicados (recorrendo aos predicados built-in *assert* e *retract*), que diferem apenas no valor do átomo Content. A imagem 4 retrata o estado do jogo após seis jogadas, sendo possível observar as posições reclamadas por cada jogador, representadas com b e g, respectivamente.

```
pos(nw, nw, 0). pos(nw, n, g). pos(nw, ne, 0).      pos( n, nw, 0). pos( n, n, 0). pos( n, ne, b).      pos(ne, nw, 0). pos(ne, n, 0). pos(ne, ne, 0).
pos(nw,  w, 0). pos(nw, c, 0). pos(nw, e,  0).      pos( n,  w, 0). pos( n, c, 0). pos( n,  e, 0).      pos(ne,  w, 0). pos(ne, c, 0). pos(ne,  e, g).
pos(nw, sw, 0). pos(nw, s, 0). pos(nw, se, 0).      pos( n, sw, 0). pos( n, s, 0). pos( n, se, 0).      pos(ne, sw, 0). pos(ne, s, 0). pos(ne, se, 0).

pos( w, nw, 0). pos( w, n, 0). pos( w, ne, 0).      pos( c, nw, b). pos( c, n, 0). pos( c, ne, 0).      pos( e, nw, 0). pos( e, n, 0). pos( e, ne, 0).
pos( w,  w, 0). pos( w, c, 0). pos( w, e,  0).      pos( c,  w, 0). pos( c, c, 0). pos( c,  e, 0).      pos( e, w, 0).  pos( e, c, 0). pos( e,  e, 0).
pos( w, sw, 0). pos( w, s, 0). pos( w, se, 0).      pos( c, sw, 0). pos( c, s, 0). pos( c, se, 0).      pos( e, sw, 0). pos( e, s, 0). pos( e, se, b).

pos(sw, nw, 0). pos(sw, n, 0). pos(sw, ne, 0).      pos( s, nw, 0). pos( s, n, 0). pos( s, ne, 0).      pos(se, nw, 0). pos(se, n, 0). pos(se, ne, 0).
pos(sw,  w, 0). pos(sw, c, 0). pos(sw, e,  0).      pos( s,  w, 0). pos( s, c, 0). pos( s,  e, 0).      pos(se, w, 0).  pos(se, c, 0). pos(se,  e, w).
pos(sw, sw, 0). pos(sw, s, 0). pos(sw, se, 0).      pos( s, sw, 0). pos( s, s, 0). pos( s, se, 0).      pos(se, sw, 0). pos(se, s, g). pos(se, se, 0).
```

Figura 4: Estado intermédio.

Como especificado nas regras acima descritas, o jogo acaba quando um dos jogadores tiver 5 mesas com pelo menos 5 peças. A figura 5 exemplifica essa

situação: o jogador verde tem 5 mesas com pelo menos 5 peças, enquanto que o jogador preto tem apenas 2, o que indica que o jogo acabou e que o vencedor é o verde.

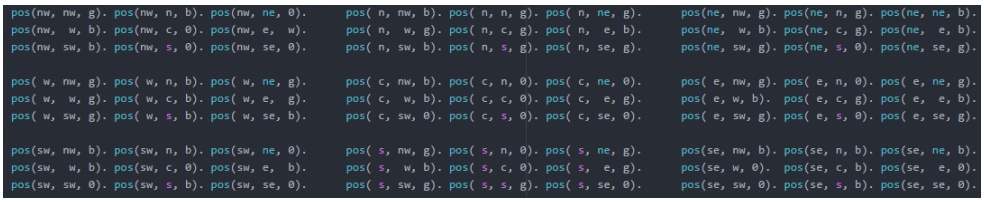

Figura 5: Estado final.

Na secção sobre a visualização do tabuleiro encontram-se representações dos três estados usando essa interface.

3 Visualização do Tabuleiro

A visualização textual do tabuleiro foi feita representando as nove mesas como matrizes de 3x3 caracteres (sem considerar espaços), sendo que cada caracter representa uma posição nessa mesa. Tanto os caracteres dentro da matriz como as próprias matrizes seguem o modelo de pontos cardeais apresentado anteriormente. As posições tomam o caracter 0 se estiverem vazias, g se pertencerem ao jogador verde e b se pertencerem ao preto, tal como usado na representação interna do estado do jogo previamente estipulada.

Visto que o estado do jogo está contido na própria base de dados do Prolog sobre a forma de predicados, não é necessário passar o estado do jogo ao predicado de visualização. Como tal, em vez de um predicado recorreu-se a um átomo, printBoard, que realiza a impressão das nove mesas linha a linha, usando um novo átomo para cada (printRow1, printRow2, etc). Estes átomos, por sua vez, mostram cada um três linhas, uma de cada mesa, recorrendo a três predicados (printTableTop(Table), printTableMiddle(Table) e printTableBottom(Table)), cada um com aridade 1 e cujo átomo é a mesa a que se referem. Finalmente, estes predicados usam o predicado printPos(Table, Pos), que imprime o conteúdo da posição Pos da mesa Table. O código que implementa este mecanismo encontra-se exemplificado na figura 7. A representação do output deste predicado usando os três diferentes estados anteriormente encontra-se na figura 6

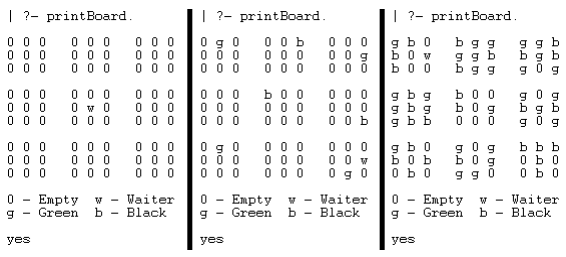

Figura 6: Visualização do tabuleiro usando, respectivamente, o estado inicial, intermédio e final.

```
printBoard :-   nl,
                printRow1, nl,
                printRow2, nl,
                printRow3, nl, nl,
                printRow4, nl,
                printRow5, nl,
                printRow6, nl, nl,
                printRow7, nl,
                printRow8, nl,
                printRow9, nl, nl,
                write('0 - Empty  w - Waiter'), nl, write('g - Green  b - Black'), nl, nl.

printRow1 :- printTableTop(nw), write('   '), printTableTop(n), write('   '), printTableTop(ne).
printRow2 :- printTableMiddle(nw), write('   '), printTableMiddle(n), write('   '), printTableMiddle(ne).
printRow3 :- printTableBottom(nw), write('   '), printTableBottom(n), write('   '), printTableBottom(ne).

printRow4 :- printTableTop(w), write('   '), printTableTop(c), write('   '), printTableTop(e).
printRow5 :- printTableMiddle(w), write('   '), printTableMiddle(c), write('   '), printTableMiddle(e).
printRow6 :- printTableBottom(w), write('   '), printTableBottom(c), write('   '), printTableBottom(e).

printRow7 :- printTableTop(nw), write('   '), printTableTop(s), write('   '), printTableTop(se).
printRow8 :- printTableMiddle(sw), write('   '), printTableMiddle(s), write('   '), printTableMiddle(se).
printRow9 :- printTableBottom(sw), write('   '), printTableBottom(s), write('   '), printTableBottom(se).

printTableTop(Table) :- printPos(Table, nw), write(' '), printPos(Table, n), write(' '), printPos(Table, ne).
printTableMiddle(Table) :- printPos(Table, w), write(' '), printPos(Table, c), write(' '), printPos(Table, e).
printTableBottom(Table) :- printPos(Table, sw), write(' '), printPos(Table, s), write(' '), printPos(Table, se).

printPos(Table, Pos) :- pos(Table, Pos, X), write(X).
```

Figura 7: Implementação do predicado de visualização do tabuleiro.

4 Movimentos

Dado que a mesa em que se pode jogar é determinada pela jogada anterior, um movimento normal apenas necessita de localizar a posição dentro da mesa. No entanto, caso nessa mesa não haja espaço para uma jogada, devem ser dados dois argumentos, um para escolher uma mesa com vagas e outro para escolher a posição. Os predicados para uma jogada podem ser, então, move/1 ou move/2, com os cabeçalhos, respetivamente, move(P) e move (T, P), em que P representa a posição e T a mesa alvo da jogada.

Para ações especiais serão definidos predicados dependendo da ação:

- specialMovePiece (T1, P1, T2, P2)
  - Usado para as ações que implicam a troca de uma peça de uma posição P1 numa mesa T1 não conquistada para qualquer posição P2 em qualquer mesa T2 não conquistada.
- specialMoveWaiter (T)
  - Aplicável nas ações que permitem a mudança do empregado para a mesma posição em qualquer mesa T.
- specialMoveRotate (N)

  - Predicado correspondente às ações especiais de rodar a mesa N posições em qualquer direção. (N é um valor entre -3 e 4, sendo que um valor positivo implica uma rotação no sentido horário).

- specialMoveSwitch (T1, T2)
  - Usado em duas ações que implicam a troca de duas mesas T1 e T2 (ambas não conquistadas ou apenas uma conquistada).

5 Fontes de Pesquisa

- `http://blackstrawgames.com/portfolio/oolong/`
- `https://boardgamegeek.com/boardgame/176212/oolong`
- `http://unpub.net/games/detail/?proto=740&o=636`