Otrio

Relatório Final

Mestrado Integrado em Engenharia Informática e Computação

Programação em Lógica

Grupo Otrio3:
Duarte Brandão - ei10060
Pedro Tavares - up201406991

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

13 de Novembro de 2016

Resumo

O objectivo deste trabalho consiste em implementar o jogo de tabuleiro Otrio, recorrendo à linguagem *Prolog*, no contexto da unidade curricular de Programação Lógica. Otrio é um jogo de tabuleiro baseado no jogo do galo em que o objectivo baseia-se em conseguir completar uma sequencia de peças do mesmo jogador para vençer. A nossa implementação permite entre 2 e 4 jogadores desafiarem-se ou um humano jogar contra 1-3 AI's.

Conteúdo

1 Introdução

O objetivo principal deste projeto prende-se em adquirir conhecimentos e experiência na área da Programação Lógica e desenvolver novas formas de abordagem a problemas.

2 Otrio

O Otrio[1] é um jogo de tabuleiro de 2 a 4 jogadores baseado no jogo do galo. É jogado num tabuleiro 3x3 (Figura 1) e cada jogador possui 3 peças de cada um dos tamanhos(Big, Medium e Small). É jogado em turnos até um dos jogadores conseguir vencer combinando as peças da sua cor numa das seguintes formas:

- Linha de 3 peças do mesmo tamanho;
- Linha por ordem crescente ou decrescente de peças;
- 3 peças dos diferentes tamanhos na mesma posição;

Figura 1: Tabuleiro de jogo

3 Lógica do Jogo

Descrever o projeto e implementação da lógica do jogo em Prolog, incluindo a forma de representação do estado do tabuleiro e sua visualização, execução de movimentos, verificação do cumprimento das regras do jogo, determinação do final do jogo e cálculo das jogadas a realizar pelo computador utilizando diversos níveis de jogo. Sugere-se a estruturação desta secção da seguinte forma:

3.1 Representação do Estado do Jogo

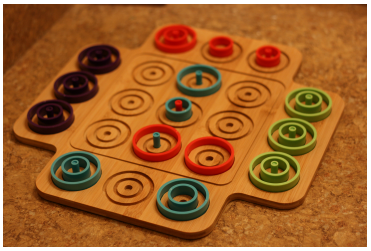
O jogo é apresentado aos jogadores na forma de uma matriz, semelhante à Figura 2. O tabuleiro é representado através de uma lista de listas, em que a lista exterior representa uma linha do tabuleiro e cada uma das listas contida nesta as 3 casas presentes numa posição do tabuleiro (Big, Medium e Small). O jogador a que a peça pertence é representado por um número após a sua casa (1 a 4), sendo o "0"usado para representar uma casa vazia.

No caso da Figura 3, o jogador 1 (peças azuis) venceu o jogo.

[1] `http://www.marblesthebrainstore.com/otrio.htm`

```
| ?- game.
[[b0,m0,s0],[b0,m0,s0],[b0,m0,s0]]
[[b0,m0,s0],[b0,m0,s0],[b0,m0,s0]]
[[b0,m0,s0],[b0,m0,s0],[b0,m0,s0]]
```

Figura 2: Estado do tabuleiro no início do jogo



(a) Tabuleiro

```
tab([[[b0,m0,s0],[b1,m0,s1],[b0,m0,s0]],
     [[b0,m0,s0],[b0,m1,s2],[b0,m0,s0]],
     [[b0,m0,s0],[b2,m0,s1],[b2,m0,s0]]
    ]).
```

(b) Estrutura do tabuleiro

Figura 3: Estado final

3.2 Visualização do Tabuleiro

Como já referenciado no estado do jogo, o nosso tabuleiro vai ter uma representação em estado de matrix 3x3 para qual será usado o predicado *printboard(+Board)*.

```
| ?- game.
[[b0,m0,s0],[b1,m0,s1],[b0,m0,s0]]
[[b0,m0,s0],[b0,m1,s2],[b0,m0,s0]]
[[b0,m0,s0],[b2,m0,s1],[b2,m0,s0]]
---
```

Figura 4: *Output* do tabuleiro da Figura 3a

3.3 Execução de Jogadas

Após um jogador introduzir os dados de uma nova jogada, este input é validado dentro do contexto de jogo, certificando-se assim que o jogador introduziu um tipo de peça e coordenadas válidas. Após este teste, é verificado se o jogador tem pelo menos uma peça do tipo escolhido na mão, usando o predicado *checkHand(+Player,+PieceSize, +Hand1,-Hand2)* e a nova peça colocada no tabuleiro de jogo conforme o tamanho da peça:

- *playBig(+Player, +Y, +X, +Board, -NewBoard)*
- *playMedium(+Player, +Y, +X, +Board, -NewBoard)*
- *playSmall(+Player, +Y, +X, +Board, -NewBoard)*

3.4 Avaliação do Tabuleiro

Após cada turno dos jogadores, o tabuleiro é avaliado pelo predicado *checkWin(+Player, +PieceSize, +Board, +Y,+X)* para verificar se a jogada permitiu ao jogador vencer e terminar o jogo.

3.5 Final do Jogo

O jogo pode terminar sem que nenhum jogador tenha vencido caso estes não tenham mais nenhuma peça na mão para jogar. Esta condição é verificada no inicio do turno do 1º jogador com o predicado *checkEndGame(+Hand,+CurrentPlayer.*

3.6 Jogada do Computador

O Computador está definido para realizar jogadas aleatórias, neste momento apenas compete gerar uma posição e caso esta seja válida executa. *playAI(Player, Tab1, Tab2, Hand1, Hand2, Size, Line, Col).*

4 Interface com o Utilizador

Para o Utilizador executar o jogo, apenas precisa de chamar o predicado *otrio()* e introduzir o modo de jogo. Após escolher o número de jogadores, o jogo é iniciado e o tabuleiro de jogo vazio é apresentado. A cada turno de um jogador é apresentado o tabuleiro com o estado do jogo e as peças que o jogador actual tem na sua mão e pode jogar. Para executar uma jogada, o utilizador tem que introduzir o tamanho da peça que deseja jogar ("big", "medium"ou "small") e as coordenadas da posição onde a deseja colocar, no formato "X-Y". Caso o input do utilizador esteja incorrecto ou a jogada seja inválida, um erro surge na consola. Após a correcta introdução da jogada, o tabuleiro é actualizado com a nova jogada e o turno passa ao próximo jogador.

5 Conclusões

Este projeto permitiu-nos aprofundar os nossos conhecimentos de Prolog e aperfeiçoar a nossa abordagem a problemas de lógica, pois esta linguagem apresenta um paradigma diferente das linguagens de programação com que tivemos contacto até à data.

Relativamente ao resultado final, consideramos que a solução foi implementada com sucesso e de acordo com os objectivos estabelecidos, embora a dificuldade da AI não seja muito elaborada e esta apenas realize jogadas válidas aleatórias.