Implementação em Prolog do Jogo Oware

Relatório Final

Universidade do Porto

Faculdade de Engenharia

FEUP

Mestrado Integrado em Engenharia Informática e Computação

Programação em Lógica

Grupo 35:
André Freitas - ei10036
Rui Gonçalves - ei10100

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

3 de Novembro de 2012

Resumo

Este relatório tem o objetivo de descrever a implementação do Jogo Oware numa linguagem de programação lógica em matemática que é o Prolog. O jogo em questão é de tabuleiro, jogando-se com sementes, sendo muito popular na República do Gana. Apesar da simplicidade das regras tem um forte componente estratégico.
O jogo foi implementado com recurso a listas para representar o tabuleiro e para representar as propriedades dos jogadores, estando organizando em módulos de manipulação do tabuleiro, rotinas de jogo, inteligência artificial, interfaces e predicados de teste. Existem 2 níveis de bots que jogam, respectivamente, aleatoriamente e com base numa heurística.

1 Introdução

Pretende-se explorar as capacidades do Prolog para representar um jogo e as suas regras através deste trabalho. Um jogo é uma excelente maneira de enriquecer o conhecimento na representação e estruturação dos dados bem como a aplicação de regras do jogo traduzidas nesta linguagem.

A implementação em questão é em modo de texto, não sendo muito refinada no sentido da apresentação mas sim na componente das funcionalidades.

Neste documento pretende-se apresentar a descrição do problema, a representação dos estados do jogo em estruturas conhecidas em Prolog, a representação das jogadas, a visualização do tabuleiro, a inteligência artificial e os aspetos de desenvolvimento do projeto.

2 Descrição do Problema

O Oware é dos jogos de tabuleiro mais antigos do mundo, tendo sido inventado há mais de 7 mil anos. É jogado por todo o Globo e não existem certezas relativamente à sua origem, porém, atribui-se a sua autoria tradicionalmente ao continente Africano. Atualmente é o jogo mais popular na República do Gana sendo um fenómeno nacional.

Como se pode constatar pela Figura 1, existe um tabuleiro com 2x6 cavidades onde se colocam sementes ou feijões. Existem muitas interpretações das regras deste jogo, pelo que iremos adotar apenas a que é mais conhecida. Assim, o jogo começa com 4 sementes em cada buraco. Os jogadores jogam alternadamente, e em cada jogada tira-se as sementes de um buraco da nossa linha de jogo e vai-se distribuindo as sementes no sentido anti-horário.

Figura 1: Pessoas jogando tradicionalmente o Oware

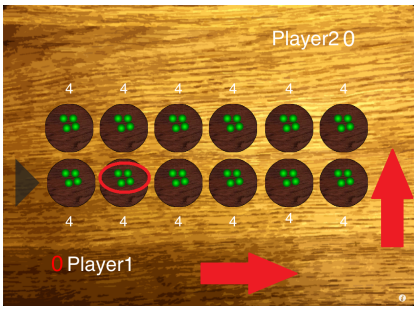
Figura 2: Distribuição das sementes

Quando distribuirmos a última semente e se colocarmos num buraco do adversário e esse sítio ficar com 2 ou 3 sementes, capturamos essas sementes. O jogo termina quando um jogador capturar 25 sementes ou ambos os dois jogadores capturarem 24 sementes (empate).

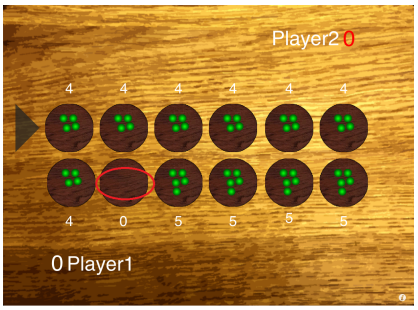
2.1 Ilustração de jogadas

Nas seguintes ilustrações serão descritas duas situações de jogos fundamentais. Pede-se especial atenção para as ilustrações, dado que é necessário estar atento às situações que descrevem para perceber o jogo.

Nesta primeira situação o jogo está a começar e o jogador vai distribuir a segunda cavidade de sementes. Assim, as sementes ficaram distribuídas e não houve captura. De notar novamente que o sentido da distribuição das sementes é anti-horário e que o jogador só pode retirar as sementes da sua linha.

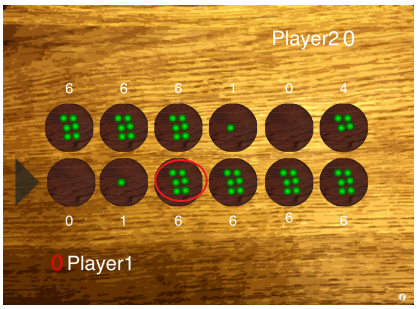


(a) Estado Inicial do Jogo

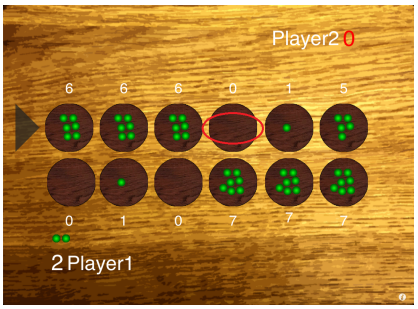


(b) Jogada sem Captura

A outra jogada importante é a da captura de sementes, ou seja, quando um jogador ao distribuir a última semente calha numa linha do seu adversário, ficam 2 ou 3 sementes nesse sítio, capturando-as. Atenção que se a meio da distribuição das sementes conseguir este número nas cavidades, não as pode capturar, por isso é só na semente final. Destaca-se esta situação dado que existem várias interpretações relativas a esta regra.



(c) Antes da Captura



(d) Após a Captura de Sementes

Pode-se constatar assim a simplicidade do jogo, mas apesar disso requer um componente estratégico para que se possa ganhar, tentando prever-se uma panóplia de jogadas possíveis para vencer o adversário e maximizar as nossas capturas.

3 Arquitectura do Sistema

Ora, como referido anteriormente, o sistema é decomposto em módulos o que permite um melhor isolamento no desenvolvimento e manutenção futura do código. Assim sendo, os módulos são os seguintes:

1. oware.pl - possui todos os predicados inerentes à rotina de jogo;
2. owareBoard.pl - contém os predicados para manipular e aceder à estrutura do tabuleiro;
3. owareCLI.pl - alberga os predicados que interagem na consola de texto com o utilizador;
4. owareAI.pl - módulo que contém os predicados para os bots;
5. owareTest.pl - módulo onde se colocam testes que sejam pertinentes;

4 Representação de um Movimento

Tal como constatado anteriormente, o jogo apenas tem uma jogada, que é a distribuição de sementes. Se dessa distribuição resultar na última posição 2 ou 3 sementes é feita uma captura. Assim sendo, existe um predicado para distribuir sementes, que por sua vez chama outro predicado que calcula a sequência de posições para distribuir, alterando depois o tabuleiro e verificando se existe possibilidade de captura através de um outro predicado. Após isto, se tiver existido uma captura, os pontos(sementes) do jogador são atualizados. Os predicados relativos a esta implementação são então os seguintes:

```
playSeeds(Board,PlayerNum,I,NewBoard)
computeSequence(PlayerNum,I,Seeds,Sequence)
removeSeeds(PlayerNum,[H|[Th|Tt]],I,Seeds,[Hnew|Tnew])
addSeeds(PlayerNum,[H|[Th|Tt]],I,Seeds,[Hnew|Tnew])
evaluateCapture(Board,WhoPlayed,I)
addScore(Score,Val,NewScore)
```

5 Visualização do Tabuleiro

Para a visualização do tabuleiro são usados três predicados: um principal para representar a totalidade do tabuleiro, um para apresentar o logótipo do Jogo e outro para imprimir uma linha de sementes.

```
printBoard([H|[Th|_]],P1Score,P2Score)
printLogo
printBoardLine([H|T])
```

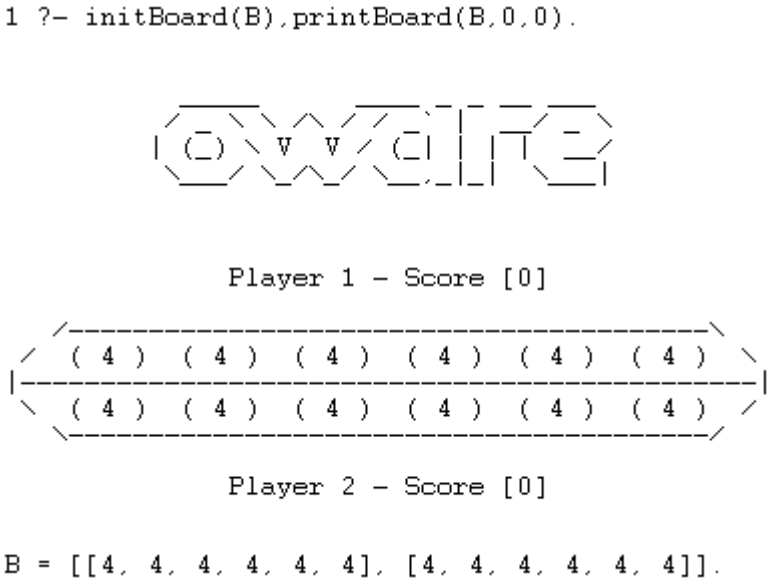


Figura 3: Começo do jogo

Estes predicados implicam sempre uma lista de listas de 6 elementos para que o tabuleiro apareça de forma correcta.

6 Conclusões e Perspetivas de Desenvolvimento

A implementação do Oware em termos de estrutura de dados é simples, porém, requer maior esforço na implementação dos predicados que envolvem a distribuição das peças, principalmente o do cálculo da sequência de posições. Existem diversas interpretações das regras e outros nomes para o Oware, por isso tentamos adotar a versão que é mais conhecida, que já foi aliás implementada em telemóveis.

Relativamente ao processo de desenvolvimento de trabalho, iremos adotar um desenvolvimento iterativo com uma metodologia ágil que é o Scrum. Assim sendo, temos o seguinte Backlog com a estimação dos pontos de esforço, estando ordenado pela prioridade dos itens:

1. Representar o tabuleiro numa estrutura [N/A]
2. Apresentar o tabuleiro em modo de texto [2 pts]
3. Criar funções básicas para listas [2 pts]
4. Deve ser possível adicionar e remover sementes na estrutura do tabuleiro [3 pts]
5. Implementar o cálculo da sequência aquando de uma distribuição de sementes [3 pts]
6. Implementar um predicado para jogar as sementes [5 pts]
7. Deve existir um predicado que avalia se há uma captura, devolvendo a posição das sementes [5 pts]
8. Implementar um predicado para ler o input do teclado [2 pts]
9. Implementar a rotina de jogo [8 pts]
10. Deve existir um bot que consegue determinar que posição das sementes deve jogar a partir do reconhecimento da linha do adversário [8 pts]

Ora assumindo este Backlog e tendo em conta que já implementamos até ao quarto item falta 80% do esforço deste trabalho, usando uma estimativa mais razoável. Tento em conta isto, não se traduz em horas de trabalho mas sim esforço e ultrapassar dificuldades. Os pontos mais críticos serão a rotina de jogo e a implementação da inteligência artificial do Bot.

7 Bibliografia

Referências

[1] The Oware Society. 2010. *Oware- Played all over the World*. Acedido a 4 de Outubro de 2012. http://www.oware.org.

[2] Awale.jpg. 2006. *A game of awale*. Acedido a 4 de Outubro de 2012. http://upload.wikimedia.org/wikipedia/commons/1/14/Awale.jpg.

[3] oware.jpg. *Playing Oware in Ghana*. Acedido a 4 de Outubro de 2012. http://exploringafrica.matrix.msu.edu/teachers/events/oware.jpg.

[4] Easy Oware 2012. *Play the classic strategy game from Africa*. Acedido a 2 de Outubro de 2012. http://itunes.apple.com/br/app/easy-oware/id408219960?mt=8.

A Código Implementado

```
% Starts the board with the default seeds
initBoard([[4,4,4,4,4,4],[4,4,4,4,4,4]]).

% Board Visualization
% Test with: initBoard(B),printBoard(B,0,0).
printBoardLine([]).
printBoardLine([H|T]):-
write(' ( '),
write(H),
write(' ) '),
printBoardLine(T).

printLogo:-
write('\n\n'),
write('           _____       ____ _ _ __ ___\n'),
write('          / _ \\ \\ /\\ / / _` |  __/ _ \\ \n'),
write('         | (_) \\ V  V / (_| | | |  __/ \n'),
write('          \\___/ \\_/\\_/ \\__,_|_|  \\___| \n\n\n').

printBoard([H|[Th|_]],P1Score,P2Score):-
printLogo,
write('\n'),
write('              Player 1 - Score ['),
write(P1Score),
write(']\n\n'),
write('   /---------------------------------------\\ \n'),
write(' / '), printBoardLine(H),write(' \\ \n'),
write('|-----------------------------------------|'),
write('\n \\ '), printBoardLine(Th), write(' / \n'),
write('   \\\---------------------------------------/\n\n'),
write('              Player 2 - Score ['),
write(P2Score),
write(']\n\n\n').


% List Core Functions
replace([_|T],0,X,[X|T]).
replace([H|T],N,X,[H|T2]):-
N>0,
N1 is N-1,
replace(T,N1,X,T2).

element([H|T],0,H).
element([_|T],N,Val):-
N>0,
N2 is N-1,
element(T,N2,Val).

elementPlus(List,I,Value,NewList):-
element(List,I,Oldval),
```

```
Newval is Oldval+Value,
replace(List,I,Newval,NewList).

elementMinus(List,I,Value,NewList):-
element(List,I,Oldval),
Newval is Oldval-Value,
replace(List,I,Newval,NewList).

% In game functions
initScore(0).
addScore(Score,Val,NewScore):-NewScore is Score+Val.

addSeeds(PlayerNum,[H|[Th|Tt]],I,Seeds,[Hnew|Tnew]):-
PlayerNum=1,
elementPlus(H,I,Seeds,Hnew),
Tnew = [Th|Tt];
PlayerNum=2,
elementPlus(Th,I,Seeds,Tnew),
Hnew = H.

removeSeeds(PlayerNum,[H|[Th|Tt]],I,Seeds,[Hnew|Tnew]):-
PlayerNum=1,
elementMinus(H,I,Seeds,Hnew),
Tnew = [Th|Tt];
PlayerNum=2,
elementMinus(Th,I,Seeds,Tnew),
Hnew = H.

% To implement later
%computeSequence(PlayerNum,I,Seeds,Sequence)
%playSeeds(Board,PlayerNum,I,NewBoard)
%evaluateCapture(Board,WhoPlayed,I)
%gameRoutine(Board,Player1Score,Player2Score):

gameRoutine(_,25,_).
gameRoutine(_,_,25).
gameRoutine(_,24,24).
```