Tafl Games

Relatório Final

Mestrado Integrado em Engenharia Informática e Computação

Programação em Lógica

Grupo Tafl2:
Duarte Brandão - 201007823
Ricardo Leite - 200902919

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

13 de Novembro de 2014

1 Resumo

O objetivo deste trabalho é implementar o jogo de Tafl, recorrendo à linguagem Prolog, no contexto da unidade curricular de Programação Lógica. Tafl pertence a uma família de jogos antigos germânicos, jogado por duas pessoas em que um dos jogadores tenta proteger e escapar do tabuleiro com o seu Rei, enquanto que o objetivo do jogador atacante é capturar o Rei do adversário. A nossa implementação permite a 2 jogadores humanos desafiarem-se um ao outro.

Conteúdo

2 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. Devido à falta de regras e registos históricos explícitos das mesmas pela antiguidade do jogo e das diversas variantes, decidimos desenvolver este projeto usando a variante Tablut por ser a melhor documentada.

3 Tafl Games

Tafl Games são uma família de jogos antigos germânicos e celtas de estratégia jogados por dois jogadores através de um tabuleiro semelhante ao que é usado no jogo do Xadrez. Os Jogos são compostos por dois exércitos, com desvantagem em número para o defensor. Existem várias variantes onde o tamanho do tabuleiro e o número de peças diferem, no entanto, todos os jogos contêm uma proporção de 2:1 em peças, onde o exército com menos peças tem a figura do Rei, que começa o jogo no centro do tabuleiro.

O objectivo do jogador defensor é chegar com a figura do Rei até uma das bordas do tabuleiro e escapar enquanto que o objectivo do exército atacante é capturar o Rei.

Figura 1: Reconstituição de um tabuleiro de Tafl

Estes jogos espalharam-se por todo o lado onde os Vikings passaram incluindo a Islândia, Grã-Bretanha, Irlanda e a região de Lapland. Algumas variantes do Tafl como Hnefatafl, Alea Evangelii, Tawlbwrdd, Brandubh, Ard Ri e Tablut foram jogadas em grande parte do Norte da Europa muito antes de 400 a.C..

Devido à sua antiguidade e suplantação do jogo pelo Xadrez na Idade Média, não existem registos completos e inequívocos das regras, sendo estas inferidas de várias fontes e relatos ao longo do tempo.

3.1 Regras

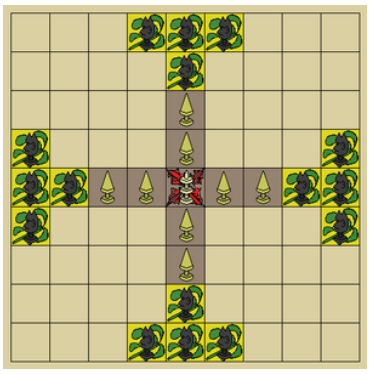
A melhor descrição encontrada até hoje foi dada por Linnaeus da variante Tablut e na qual nos vamos basear para desenvolver este jogo.

- Utiliza-se um tabuleiro 9x9 com a preparação do mesmo igual à apresentada na Fig. 2.
- O Rei começa o jogo no centro do tabuleiro.
- Os oito defensores colocam-se adjacentes ao Rei, em forma de cruz.
- Os dezasseis atacantes formam grupos de 4 no centro de cada lado do tabuleiro.
- Todas as restantes posições neutras podem ser ocupadas por qualquer peça.
- Qualquer peça pode-se mover qualquer número de casas em qualquer linha recta, mas não diagonalmente. (Semelhante à torre no Xadrez)
- Nenhuma peça pode passar por cima de outra.
- Se o Rei tiver um caminho desimpedido para a borda do tabuleiro, a não ser que seja imediatamente impedido por um atacante, este pode escapar e o jogador defensor vence.
- Qualquer peça pode ser capturada e removida de jogo se esta se encontrar rodeada por inimigos em 2 lados opostos.
- Caso o Rei seja capturado, o jogador atacante vence.

4 Lógica do jogo

4.1 Representação e Visualização do Estado do Jogo

O jogo é representado na forma de uma matriz, semelhante a Figura 2. As peças do jogador atacante são representadas por 'b' e as peças do jogador defensor por 'w', sendo a figura do Rei representada por 'k'. As casas vazias são representadas por '-'. Internamente, esta representação é guardada numa lista de listas, para facilitar o seu manuseamento e utilização.



(a) Tabuleiro a representar

```
| ?- game.
0 1 2 3 4 5 6 7 8 9
1 - - - b b b - - -
2 - - - - b - - - -
3 - - - - w - - - -
4 b - - - w - - - b
5 b b w w k w w b b
6 b - - - w - - - b
7 - - - - w - - - -
8 - - - - w - - - -
9 - - - b b b - - -
b, choose the piece to move:(x-y)
|:
```

(b) Representação inicial

```
| ?- game.
0 1 2 3 4 5 6 7 8 9
1 - - - b b b - - -
2 - - - b b - - - -
3 - k b - w w - - -
4 b w - - w - - - b
5 b - - w - - - b b
6 b - - - w - - - b
7 - - - - w - w - -
8 - w - - - - - - -
9 - - - b b b - b -
b, choose the piece to move:(x-y)
|:
```

(c) Representação a meio de um jogo

Figura 2: Interface com os jogadores

Os predicados responsáveis pela representação no ecrã são o *printSBoard*,*printSideLine*, *printtopline* e o *printLine*, que se encontram documentados nos Anexos.

4.2 Movimentos

As peças apenas se podem mover na vertical ou na horizontal, da mesma maneira que a torre no xadrez, nunca podendo passar por cima de outras peças. Sendo assim, cada um dos jogadores terá que escolher que peça deseja movimentar e o seu local de destino, sendo depois avaliado se a jogada é válida e efetuado o movimento.

Ao jogador é perguntado a posição da peça que deseja mover. no formato (x-y), e a posição final que deseja que esta ocupe, também inserida no mesmo formato. Após a inserção de cada par de valores, estes são validados, isto é, se a posição correspondente a peça que se deseja mover pertence ao jogador em questão e se a posição final da peça corresponde a uma casa vazia. Caso alguma das posições não seja valida, o jogador deve inserir umas novas coordenadas validas.

Os predicados responsáveis pelo tratamento e validação das coordenadas são:

- *choosePiece* e *chooseDest*, responsável pelo prompt ao utilizador
- *getPiece*, trata o input do jogador
- *validatePiece*, que verifica que a peça que se encontra na posição especificada é uma peça pertencente ao jogador ou uma posição não ocupada.

Após a validação da posição inicial e final, verifica-se se o caminho entre as duas posições é valido, ou seja, se não existem peças entre as posições. Esta verificação é feita pelo predicado *validMove*.

Após estas verifições, o movimento em si é realizado, sendo a matriz actualizada para reflectir as alterações.

4.3 Captura de Peças

Qualquer peça é capturada se se encontrar entre 2 peças inimigas. Sendo assim, após cada movimentação, é verificado se existe uma peça inimiga em qualquer um dos pontos cardeais. Caso exista, é verificado se a posição a seguir a essa é uma peça aliada, sendo que a peça inimiga se encontra em posição de ser capturada.

As funções responsáveis pela captura são *captures* e *capture*.

4.4 Fim do Jogo

No fim de cada movimento, é verificado se o rei ainda se encontra em jogo e não foi capturado. Caso este não se encontre no tabuleiro, quer dizer que o jogador atacante venceu. Caso a peça do rei se encontre numa borda, ou seja, uma posição em que a coordenada X=1 ou X=9 ou então Y=1 ou Y=9, o Rei consegue escapar e vence o jogo. Caso nenhuma destas condições se verifique, o jogo continua normalmente.

5 Conclusão

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 do projeto foi satisfatório, sendo que faltou implementar um modo de jogo entre 2 AI e entre um jogador e uma AI.

Referências

[1] http://en.wikipedia.org/wiki/Tafl_games