Xadrez Massacre

Relatório Final

Mestrado Integrado em Engenharia Informática e Computação

Programação em Lógica

Grupo 03:
Pedro França Santos - ei12056
Pedro Miguel Ferreira - ei11109

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

12 de Novembro de 2017

Resumo

Este relatório diz respeito ao primeiro trabalho prático da unidade curricular de Programação em lógica, que visa desenvolver em linguagem PROLOG o jogo de tabuleiro Xadrez Massacre.

Durante o desenvolvimento deste trabalho pudémos aplicar e por em prática alguns dos conhecimentos que temos vindo a interiorizar durante as aulas teóricas e teórico-práticas desta unidade curicular ao long do semestre.

Podemos dizer que este trabalho por muitas vezes nos levou à exaustão derivado da linguagem ter um paradigma completamente diferente dos que estamos acostumados sendo PROLOG uma Linguagem Lógica baseada no conceito de Robert Kowalski da interpretação procedimental das cláusulas de Horn, e que portanto requer um raciocínio bastante diferente das Linguagens Funcionais e Programação Orientada a Objectos a que estamos habituados. Ainda assim, com a ajuda de pesquisas, de consulta dos materias e exercicos resolvidos das aulas, de predicados já existente e de outros por nós criados conseguimos ir resolvendo cada problema com que nos deparamos.

No entanto temos a lamentar não ter cumprindo a totalidade dos objectivos propostos, derivado da falta de tempo, pois o grande numero de problemas com que nos deparamos e o tempo necessario para os resolver não foi compativel com a altura do semestre actual e a necessidade de trablhar também para outras unidades curriculares. Ainda assim temos a funcionalidade humano contra humano a funcionar na totalidade com toda a logica de jogo implementada, carecendo o trabalho das funcionalidades de humano vs. Computador bem como Computador vs. Computador.

Conteúdo

1 Introdução

No ambito da unidade curricular de PLOG, do curso Mestrado Integrado em Engenahria Informatica e Computação proposemo-nos a elaborar o jogo Xadrez Massacre na linguagem PROLOG que nos apresenta um novo paradigma de programação. O objectivo deste trabalho foi a aplicação dos primeiros conceitos

interiorizados nas aulas de unidade curricular.Este método de avaliação torna-se importante pois permite-nos avaliar os conhecimentos que adquirimos até então e perceber se somos ou não capazes de trabalhar com PROLOG. Este relatorio

encontra-se dividido em várias secções tais como:

- Regras e jogadas possiveis do Xadrez Massacre.
- A lógica de jogo, descrição do projecto e da sua implementação em PROLOG
- O modo de interação entre o utilizador e o programa, jogabilidade e visualização.
- Conlusão final do grupo a este primeiro desafio com a linguagem PROLOG.
- Lista de referências.

2 O Jogo Xadrez Massacre

Xadrez Massacre é um jogo para dois jogadores criado por Andy Lewicki. Cada jogador começa com 8 Rainhas, 8 Torres, 8 Bispos e 8 Cavalos que se movimentam de acordo com as regras clássicas do xadrez. As 64 peças são colodas inicialmente no tabuleiro de forma aleatória.

Figura 1: Possivel configuração de um tabuleiro inicial

Inicio do jogo:
Por convenção começam a jogar as Brancas, no entanto pode ser usado qualquer outro método para decidir quem começa a jogar. Na nossa abordagem as brancas começam.

Regras do jogo:

É obrigatorio capturar uma peça do adversario em cada jogada. Não são validas quaisquer outras jogadas.

O jogador que não conseguir fazer uma jogada perde o jogo.

O jogador que perder todas as peças também perde o jogo.

Figura 2: Possivel estado final em que a rainha branca capturou de E7 para E8 e as pretas não têm movimentos possivéis.

3 Lógica do Jogo

3.1 Representação do Estado do Jogo

O estado do jogo é representado por uma lista de quatro elementos onde é armazenada toda a imformação do estado do jogo, criando assim uma espécie de classe (mais uma vez, derivado da nossa familiaridade com a programação orientada a objectos).

O primeiro elemento da lista é uma lista de listas que representam o estado actual do tabuleiro de jogo, ou seja, o conteudo de cada posição no tabuleiro. A disposição das peças no tabuleiro é, portanto, armazenada numa lista de listas (matriz).

O segundo elemento da lista é outra lista de dois elementos cujo conteudo é, respectivamente, o numero de peças que o jogador branco e o jogador preto têm sobre o tabuleiro.

O terceiro elemento determina qual dos jogadores deve efectuar a jogada naquele estado. Para um estado de jogo em que seja a vez do jogador branco efetuar uma jogada, o terceiro elemento será portanto 'whitePlayer', por sua vez, se fosse o jogador preto o próximo a jogar, o terceiro elemento seria 'blackPlayer'.

Finalmente, o quarto elemento contem o modo de jogo. Humano contra humano(pvp), humano contra computador(pvc) e computador contra computador(cvc). Apesar de esta ser a intenção inicial, por falta de tempo apenas se encontra implementada a opção humano contra humano(pvp) no entanto se tivermos possibilidade, mesmo depois da entraga, iremos tentar implementar os restantes modos de jogo, até para consolidar conhecimentos.

3.2 Visualização do Tabuleiro

O tabuleiro de jogo é representado em prolog por uma lista de listas, neste caso uma lista de 8 listas com 8 elementos cada uma.

Para a lógica do programa usamos internamente as letras "q" para representar uma dama (queen), "t" para representar uma torre, "b" para representar um bispo e "h" para representar um cavalo (horse). Usamos tambem as letras "w" e "b" para representar se uma peça é respectivamente branca ou preta (white or black). Para representar espaço em branco (' ') usamos internamente "emptyCell".

Para display no tabuleiro em consola, usamos as mesmas letras para representar as peças mas usando letra maiuscula para as preta e minusculas para as brancas.

Exemplo de representação de um estado inicial do tabuleiro:

```
[[bq, bq, bh, bb, bq, bt, bb, wb],
[wq, wt, wb, wh, wq, bt, bb, bh],
[bh wt, wb, bq, bq, wb, bb, wq],
[bq, wt, wb, wh, wt, bt, wh, wt],
[wq, bb, wh, wb, bq, bt, wt, bh],
[bt, wt, wb, wh, wq, wq, bb, bh],
[wq, bh, wb, bt, bq, wh, bb, wt],
[wq, bh, bt, wh, wh, bb, bt, bh]].
```

Os predicados responsáveis pela visualização do tabuleiro na linha de comandos encontram-se no final do ficheiro 'XadrezMassacre.pl'. Para imprimir o tabuleiro basta chamar 'printBoard(Board)' onde 'Board' é um tabuleiro de um estado de jogo. Por sua vez, este preicado faz uso de outros perdicados com funções mais especificas para imprimir o tabuleiro de forma formatada.

3.3 Lista de Jogadas Válidas

Após escolher selecionar a peça que quer mover é apresentada ao jogador uma lista de todas as jogadas válidas para a peça selecionada. A forma como são calculadas essas jogada está explicada na secção 3.4 em baixo...

3.4 Execução de Jogadas

Em cada jogada, é pedido ao jogador que tem a vez de jogar as coordenadas da peça no tabuleiro que deseja movimentar. Logo depois de as coordenadas serem validadas, o programa verifica se as coordenadas correspondem a uma peça do jogador que tem a vez de jogar e não a uma das peças do oponente ou uma 'emptyCell'. Esta verificação é feita com recurso ao predicado ***validateChosenPieceOwnership(SrcRow,SrcCol,Board,Player)*** e caso falhe, o programa retrocede e pede ao jogador para inserir novas coordenadas.

Uma vez selecionada a peça pretendida o programa calcula todas as possiveis jogadas para essa peça e guarda-as na base de dados ***assert(coords))*** em que (***dynamic:-coords/1.))*** é um predicado dinamico que pode ser alterado em

runtime) e armazena as coordenas de todas as movimetações possiveis. É então pedido ao utilizador para introduzir as coordenas de destino que pretende para a peça selecionada que são comparadas com todas as coordenadas existentes em *coords)* e se existirem lá é então uma jogada válida procedendo-se assim à captura e fazendo update ao *Board)*.

Para o calculo de todas as possiveis movimentações são usados diversos perdicados conforme a peça selecionada (***checkTower(SrcRow,SrcCol,Board,Player)***, ***checkBispe(SrcRow,SrcCol,Board,Player)***, ***checkQueen(SrcRow,SrcCol,Board,Player)***, ***checkHorse(SrcRow,SrcCol,Board,Player)***) usando a lógica de jogo e as regras clássicas do xadrez.

Finalmente, depois de a jogada completa ter sido efetuada, o predicado ***changePlayer(TempGame, ResultantGame)*** trata de alternar os jogadores entre jogadas e obter o resultado final do estado de jogo.

3.5 Avaliação do Tabuleiro

Após cada jogada e entes de imprimir o novo tabuleiro os perdicados ***assertBothPlayersHavePiecesOnTheBoard(Game)*** que verifica se algum dos jogadores deixou de ter peças, bem como o ***assertCurrentPlayerCanMove(Game)*** que percorre todas as celulas do tabuleiro e verifica se alguma das peças do jogador actual têm jogadas possiveis. Se algum destes perdicados falhar o jogo acaba.

3.6 Final do Jogo

Usando as funcionalidades e predicados referidos no ponto acima é verificada após cada jogada a condição final de jogo. Se algum destes falar entra no perdicado ***endGame(Game)*** que acaba o jogo e faz display qual jogador ganhou.

3.7 Jogada do Computador

Por falta de tempo, infelizmente não foi implementada esta funcionalidade.

4 Interface com o Utilizador

A interface na linha de comandos foi feita de forma a ser simples para o utilizador utilizar e jogar o nosso jogo Xadrez massacre. Em baixo apresentamos a interface em consola de diversas fases do jogo visualizadas.

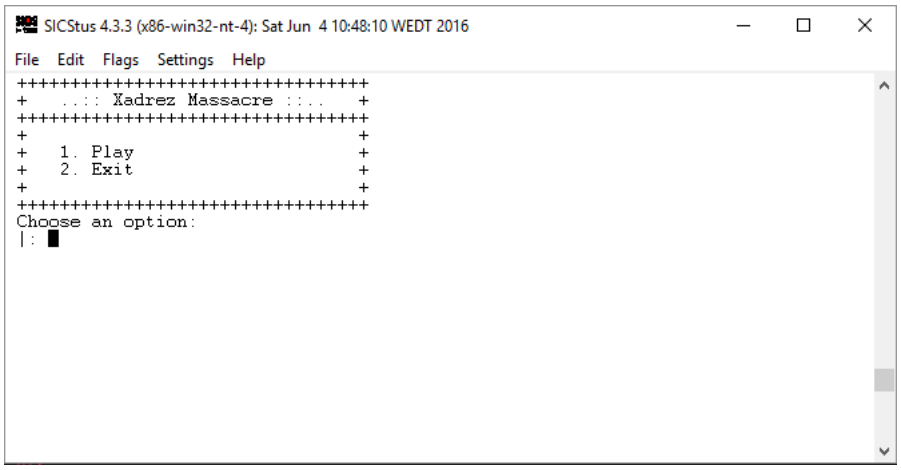

Figura 3: Menu Inicial.

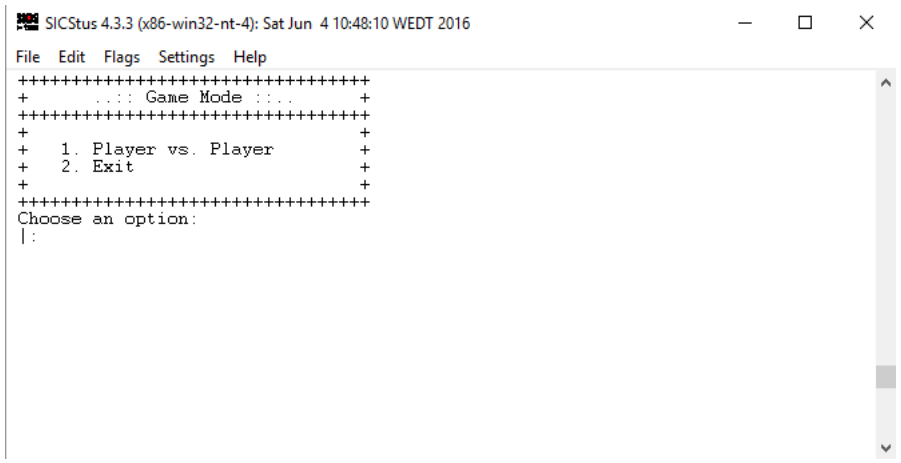

Figura 4: Escolha do modo de jogo.

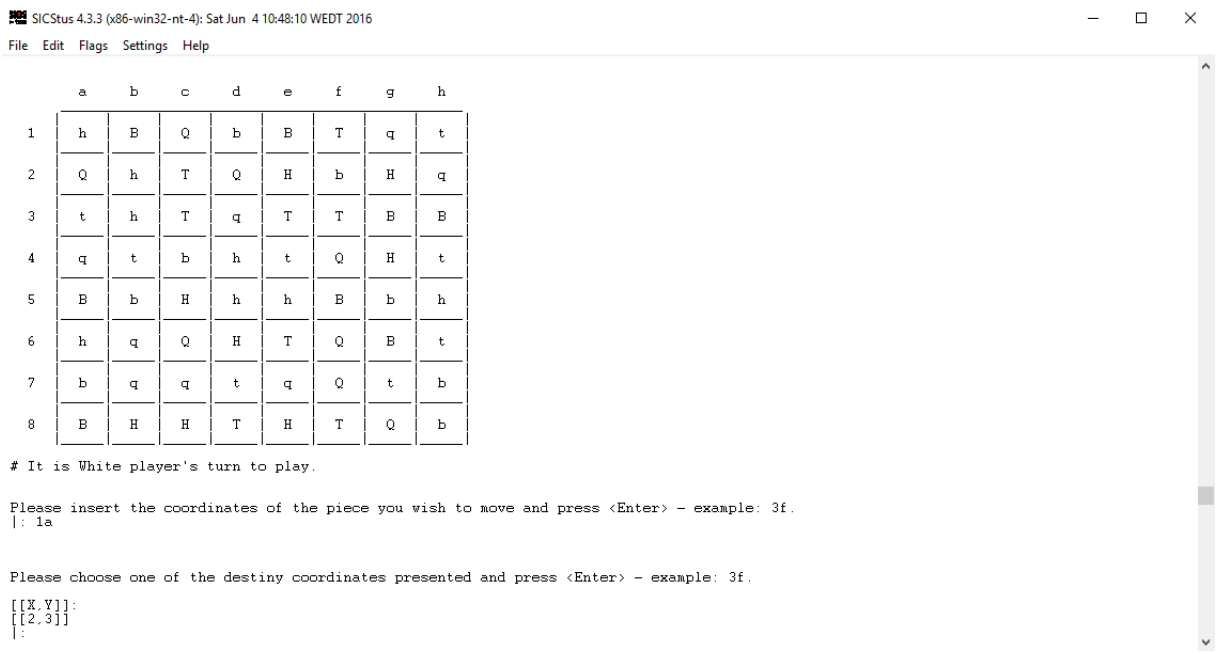

Figura 5: Tabuleiro inicial, depois de escolhida a peça a movar e onde é pedido input do destino.

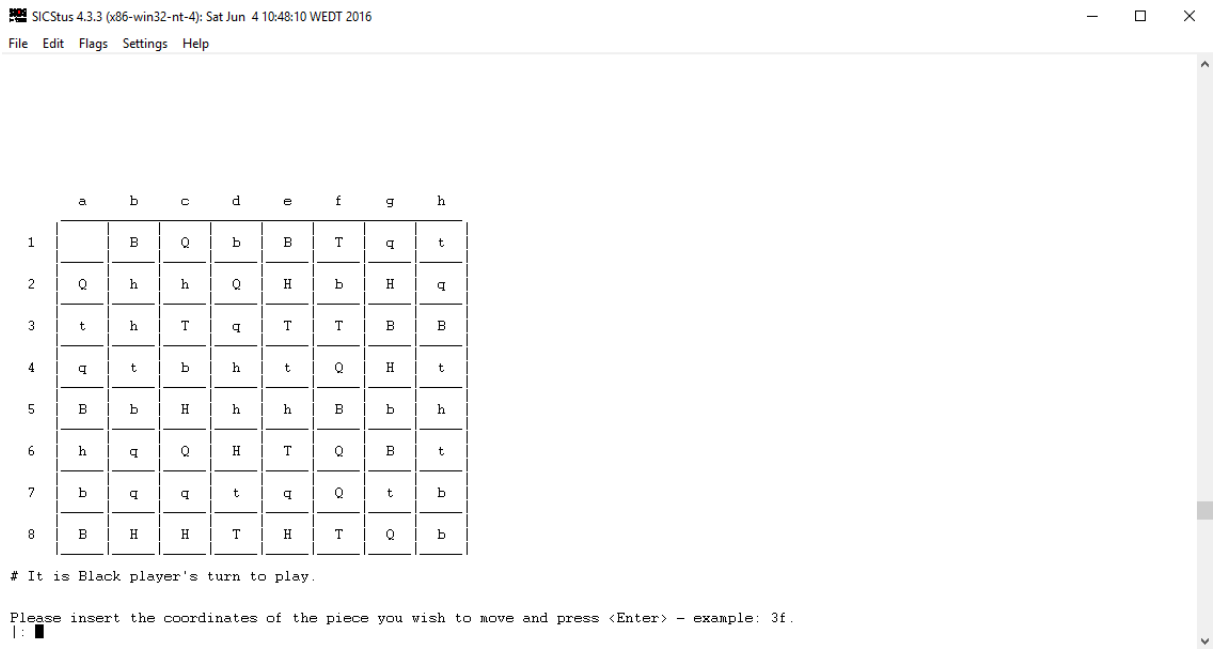

Figura 6: Apos ter capturado a peça em cima, são agora pedidas as coordenadas ao outro jogador.

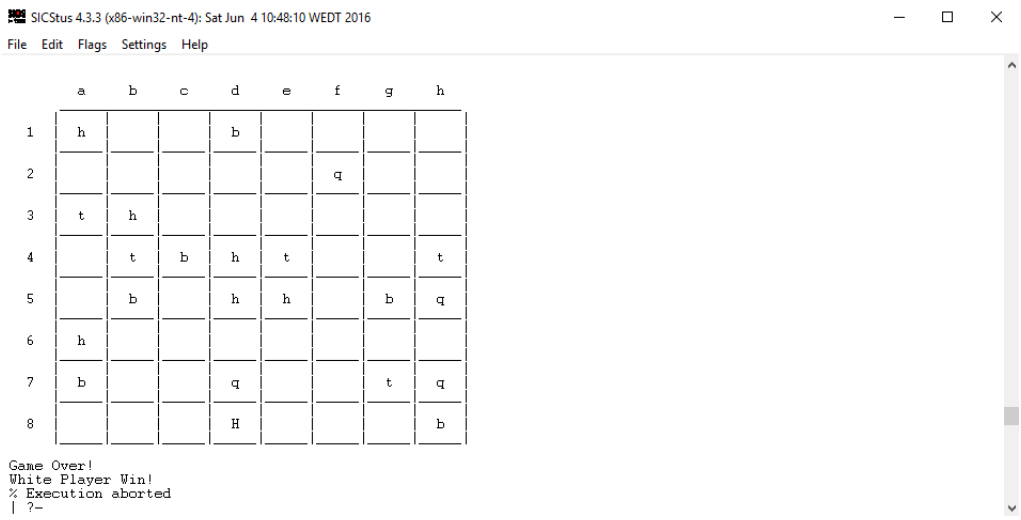

Figura 7: Exemplo de um estado final de jogo.

5 Conclusões

Inicialmente o grupo não esperava que a execução de este trabalho fosse um processo tão demorado, nem que iriamos encontrar tantas dificuldades em mudar o nosso método de pensamento já tão enraizado daí não termos concluido todos os objectivos.

No entanto temos orgulhos nos nossos resultados e sabemos que adquirimos bastantes conhecimentos durante o desenvolvimento do projecto.Apesar do escasso tempo para a entrega final do prohecto e as consecutivas sobreposições de trabalhos de outras unidades curriculares, conseguimos implementar um modo Humano vs. Humano do qual temos algum orgulho.

Xadrez Massacre revelou-se um desafio interessante e consideramos e o maior desafio em termos de programação foi cumprido. As dificuldades encontradas foram superadas, no entanto podem haver melhorias, e implementação da parte em falta.

Em suma o grupo concorda que este foi provavelmente um dos trabalhos mas desafiadores que já teve pois ao desenvolver em linguagem PROLOG tudo requer um pensamento que não é o que estamos habituados.

Bibliografia

http://www.swi-prolog.org/

http://cs.union.edu/ striegnk/learn-prolog-now/html/

https://stackoverflow.com/

Documentos fornecidos na página Moodle da unidade curricular de Programação em Lógica

A Anexo

Código Prolog implementado devidamente comentado entregue em anexo.