Projeto Nº 2: Época Normal

Inteligência Artificial - Escola Superior de Tecnologia de Setúbal
2023/2024

Prof. Joaquim Filipe
Eng. Filipe Mariano

1. Jogo do Cavalo: Descrição Geral

O Jogo do Cavalo é uma variante do problema matemático conhecido como o Passeio do Cavalo, cujo objetivo é, através dos movimentos do cavalo, visitar todas as casas de um tabuleiro similar ao de xadrez. Esta versão decorrerá num tabuleiro de `10` linhas e `10` colunas (10x10), em que cada casa possui uma pontuação. Nesta secção pretende-se dar uma perspetiva geral do que é este jogo, deixando para a secção seguinte a explicação do que se pretende que seja desenvolvido no projeto de Inteligência Artificial relativo à resolução de um problema neste contexto por procura em Espaço de Estados.

	A	B	C	D	E	F	G	H	I	J	
1	92	52	04	33	15	20	30	23	82	94	1
2	18	51	22	08	35	59	69	61	36	49	2
3	91	12	74	93	11	05	19	75	28	21	3
4	26	88	50	99	37	64	47	97	98	62	4
5	44	83	54	41	43	76	67	09	96	79	5
6	32	06	34	48	27	16	85	68	60	07	6
7	55	87	72	58	81	40	17	73	38	95	7
8	00	29	70	42	13	10	80	78	14	24	8
9	56	01	71	39	84	77	03	25	63	66	9
10	65	90	89	45	46	02	57	31	86	53	10
	A	B	C	D	E	F	G	H	I	J	

Figura 1: Exemplo de tabuleiro inicial.

1.1. Tabuleiro

O jogo possui as seguintes características:

- O tabuleiro tem as dimensões 10x10 em que os valores de cada casa são entre `00` e `99`, sem repetição.
- Cada vez que é iniciado um jogo é construído um novo tabuleiro com o valor das casas distribuído aleatoriamente.

- O objectivo do jogo é acumular mais pontos que o adversário, usando um cavalo de xadrez. Cada jogador tem um cavalo da sua cor (branco ou preto).

1.2. Desenrolar do Jogo

- O jogo do cavalo joga-se com 2 jogadores em que cada jogador possui uma peça do xadrez tradicional, que é o cavalo. O Jogador 1 joga com o cavalo branco e o Jogador 2 joga com o cavalo preto.
- Sempre que se inicia uma partida, o tabuleiro é gerado com o valor das casas distribuído aleatoriamente.
- O jogo começa com a colocação do **cavalo branco na casa de maior pontuação da 1ª linha (A1-J1 do tabuleiro)**.
- Se a casa escolhida tiver um número com dois dígitos diferentes, por exemplo 57, então, em consequência, o número simétrico 75 é apagado do tabuleiro, tornando esta casa inacessível durante o resto do jogo. Ou seja, nenhum cavalo pode terminar outra jogada nessa casa.
- Se um cavalo for colocado numa casa com um número "duplo", por exemplo 66, então qualquer outro número duplo pode ser removido e o jogador deve escolher qual em função da sua estratégia (por *default* remover a de maior valor). Depois de um jogador deixar a casa para se movimentar para outra, a casa onde estava fica também inacessível para o jogo, ficando o numero da casa apagado.
- Após a primeira jogada (colocar o cavalo branco) segue-se a jogada do adversário, com colocação do **cavalo preto na casa de maior pontuação da 10ª linha (A10-J10)** do tabuleiro, casa essa que é escolhida pelo 2º jogador. Após a colocação do cavalo preto, aplica-se também a regra do número simétrico ou do número duplo.
- Depois de ambos os cavalos serem colocados, todas as jogadas seguintes são efectuadas através de um movimento de cavalo (usando as regras tradicionais do Xadrez para o cavalo). **Um cavalo não pode saltar para uma casa vazia (sem número) e também não pode fazê-lo para uma casa que esteja ameaçada pelo cavalo adversário.**
- A cada jogada de um jogador repete-se a regra do simétrico ou duplo.
- Um jogador ganha pontos por cada casa visitada pelo seu cavalo (igual ao valor da casa). Os pontos são contabilizados apenas para as casas visitadas, não pelos números simétricos ou duplos removidos.
- Caso um dos jogadores não consiga movimentar o seu cavalo, cede a vez ao jogador oposto.

1.3. Determinar o Vencedor

O jogo termina quando não for possível movimentar qualquer um dos cavalos no tabuleiro, sendo o vencedor o jogador que ganhou **mais** pontos.

2. Objetivo do Projeto: Versão Dois Jogadores

2.1. Objetivo

No âmbito deste projeto vamos considerar o Jogo do Cavalo na versão de dois jogadores, possibilitando um enquadramento teórico– prático com os conhecimentos adquiridos no âmbito da Teoria de Jogos. Neste caso aplicam–se as regras de jogo descritas na Secção 1.

Tal como na 1ª fase do Projecto, pretende–se que os alunos desenvolvam um programa, em `Lisp`. O programa deverá implementar o algoritmo AlfaBeta ou Negamax com cortes alfa–beta, e as funções auxiliares que permitirão realizar as partidas do jogo.

Pretende–se que o programa permita ao computador vencer o jogador humano ou um outro computador, pelo que deverá funcionar em dois modos:

1. Humano vs Computador
2. Computador vs Computador

No modo 1, no início de cada partida, o utilizador deve decidir quem começa, humano ou computador, e qual o tempo limite para o computador jogar, enquanto que no modo 2 apenas é necessário definir o tempo limite.

O jogo é iniciado pelo Jogador 1 colocando o cavalo branco na casa de maior valor na 1ª linha do tabuleiro. O processo é análogo para o Jogador 2, que coloca o cavalo preto na casa de maior valor da 10ª linha do tabuleiro. A cada jogada, mediante a casa em que se colocou o cavalo, é aplicada a regra do número simétrico ou a regra do número duplo.

Caso um dos jogadores não consiga movimentar o cavalo, cede a vez ao jogador oposto. O jogo termina quando nenhum jogador consegue movimentar o cavalo.

2.2. Funcionamento

No início de cada partida, o utilizador deve decidir quem começa, humano ou computador, e qual o tempo limite para o computador jogar, o qual será de `X` milissegundos. O valor de `X` deverá ser um valor compreendido entre 1000 e 5000 milissegundos.

A jogada do jogador computador é encontrada através da aplicação do algoritmo de jogo, o AlfaBeta ou Negamax com cortes alfa–beta..

No que diz respeito à jogada do jogador humano, o programa deverá ler a jogada inserida através do teclado (coordenadas da linha e coluna), apresentar a sua própria jogada no ecrã e repetir o procedimento até a partida terminar. O programa deverá ainda ir escrevendo num ficheiro `log.dat` o número de nós analisados, o número de cortes efetuados (de cada tipo), o tempo gasto em cada jogada e o tabuleiro atual.

3. Formulação do Problema

3.1. Tabuleiro

O tabuleiro é representado sob a forma de uma lista composta por `10` outras listas, cada uma delas com `10` átomos. Cada uma das listas representa uma linha do tabuleiro, enquanto que cada um dos átomos possui o valor da respetiva casa. Por outras palavras, o tabuleiro é representado por uma lista de listas em LISP, sendo que cada átomo representa uma casa em que a linha corresponde ao índice da sublista e a coluna ao índice do átomo dentro da linha. Cada célula do tabuleiro poderá ser representada pelo seguinte valor:

- De `00` a `99` significa que a casa ainda não foi visitada;
- `NIL` significa que a casa já foi visitada;
- `-1` significa que o cavalo branco (Jogador 1) se encontra nessa casa;
- `-2` significa que o cavalo preto (Jogador 2) se encontra nessa casa.

3.2. Estrutura do Programa

O programa deverá estar dividido em três partes, cada uma num ficheiro diferente:

1. Uma parte para o algoritmo AlfaBeta ou Negamax (`algoritmo.lisp`).
2. Outra que deve conter as funções que permitem escrever e ler em ficheiros e tratar da interação com o utilizador (`interact.lisp`).
3. E a terceira parte corresponde aos operadores do jogo (`jogo.lisp`).

Enquanto que a primeira parte do programa deverá ser genérica para qualquer jogo que recorre ao algoritmo AlfaBeta ou Negamax com cortes alfa–beta (independente do domínio de aplicação), a segunda e a terceira parte são específicas do Jogo do Cavalo (dependente do domínio de aplicação).

Na parte 2 deverá ser definida uma função com o nome `jogar` com os parâmetros `estado` e `tempo`, e que devolva uma lista com o tabuleiro em que o deverá ser feita a próxima jogada.

4. Grupos

Os projetos deverão ser realizados em grupos de, no máximo, duas pessoas sendo contudo sempre sujeitos a avaliação oral individual para confirmação da capacidade de compreensão dos algoritmos e de desenvolvimento de código em Lisp.

Os grupos deverão ser os mesmos que realizaram a 1ª fase do Projeto, seguindo as regras anteriormente estipuladas em que deve ser constituido por alunos que frequentam a mesma turma e, apenas em casos excepcionais e com aprovação dos docentes, é que poderão existir grupos com elementos provenientes de turmas diferentes.

5. Datas

Entrega do projeto: 26 de Janeiro de 2024, até às 23:00.
Discussão do projeto: Inicio de Fevereiro de 2024.

6. Documentação a Entregar

A entrega do projeto e da respetiva documentação deverá ser feita através do Moodle, na zona do evento "Entrega do 2º Projeto". Todos os ficheiros a entregar deverão ser devidamente arquivados num ficheiro comprimido (`ZIP` com um tamanho máximo de 5Mb), até à data acima indicada. O nome do arquivo deve seguir a estrutura `nomeAluno1_numeroAluno1_nomeAluno2_numeroAluno2_P2`.

6.1. Código Fonte

Os ficheiros de código devem ser devidamente comentados e organizados da seguinte forma:

interact.lisp Carrega os outros ficheiros de código, escreve e lê de ficheiros e trata da interação com o utilizador.

jogo.lisp Código relacionado com o problema.

algoritmo.lisp Deve conter a implementação do algoritmo de jogo independente do domínio.

6.2. Manuais

No âmbito da Unidade Curricular de Inteligência Artificial pretende–se que os alunos pratiquem a escrita de documentos recorrendo à linguagem de marcação ***Markdown***, que é amplamente utilizada para os ficheiros

ReadMe no ***GitHub***. Na Secção 4 do guia de Laboratório nº 2, encontrará toda a informação relativa à estrutura recomendada e sugestões de ferramentas de edição para ***Markdown***.

Para além de entregar os ficheiros de código e de problemas, é necessário elaborar e entregar 2 manuais (o manual de utilizador e o manual técnico), em formato `PDF` juntamente com os sources em `MD`, incluídos no arquivo acima referido:

ManualTecnico.pdf O Manual Técnico deverá conter o algoritmo implementado, devidamente comentado e respetivas funções auxiliares; descrição dos tipos abstratos usados no programa; identificação das limitações e opções técnicas. Deverá ser apresentada uma análise crítica dos resultados das execuções do programa, onde deverá transparecer a compreensão das limitações do projeto. Deverão fazer uma análise estatística acerca de uma execução do programa contra um adversário humano, mencionando o limite de tempo usado e, para cada jogada: o respetivo valor, a profundidade do grafo de jogo e o número de cortes efetuado no processo de análise. Poderão utilizar os dados do ficheiro `log.dat` para isso.

ManualUtilizador.pdf O Manual do Utilizador deverá conter a identificação dos objetivos do programa, juntamente com descrição geral do seu funcionamento; explicação da forma como se usa o programa (acompanhada de exemplos); descrição da informação necessária e da informação produzida (ecrã/teclado e ficheiros); limitações do programa (do ponto de vista do utilizador, de natureza não técnica).

7. Avaliação

Tabela 1: Grelha de classificação.

Funcionalidade	**Valores**
Algoritmo AlfaBeta ou Negamax com Cortes alfa–beta e determinar Jogada do Computador	5
Jogada Humano	1
Apresentar Estatísticas a cada Jogada (ecrã e ficheiro)	2
Função de Avaliação	2
Implementação do limite de tempo para o computador	1
Procura quiescente	1
Memoização	1
Operadores do jogo	2
Ordenação dos nós	1
Qualidade do código	2
Manuais (utilizador e técnico)	2
Total	20

A avaliação do projeto levará em linha de conta os seguintes aspectos:

- Data de entrega final – Existe uma tolerância de 3 dias em relação ao prazo de entrega, com a penalização de 1 valor por cada dia de atraso. Findo este período a nota do projeto será 0.

- Correção processual da entrega do projeto – (Moodle; manuais no formato correto). Anomalias processuais darão origem a uma penalização que pode ir até 3 valores.
- Qualidade técnica – Objetivos atingidos; Código correto; Facilidade de leitura e manutenção do programa; Opções técnicas corretas.
- Qualidade da documentação – Estrutura e conteúdo dos manuais que acompanham o projeto.
- Avaliação oral – Eficácia e eficiência da exposição; Compreensão das limitações e possibilidades de desenvolvimento do programa. Nesta fase poderá haver lugar a uma revisão total da nota de projeto.

Campeonato

Após a entrega dos projetos, será realizado um campeonato entre os programas de cada grupo (os alunos deverão manifestar a sua intenção de participar, fazendo um registo no Moodle por ocasião da entrega do projeto 2). Para participar, será necessário que:

a) o programa esteja totalmente inserido num package com a designação **p** em que é o número do grupo registado no Moodle na grelha das séries avaliadas.
b) seja definida uma função com o nome `jogar` com os parâmetros `estado` e `tempo` (em milissegundos) e que devolva uma lista com o novo estado.

Todos os participantes neste campeonato recebem um bónus na nota final do projeto 2, com destaque para os dois primeiros lugares:
a) 3 valores para o grupo vencedor.
b) 2 valores para o grupo que acabar na 2ª posição.
c) 1 valor para os restantes grupos que se inscreverem no campeonato e em que o programa esteja corretamente estruturado para participar.

8. Recomendações Finais

Com este projeto pretende–se motivar o paradigma de programação funcional. A utilização de variáveis globais, de instruções de atribuição do tipo set, setq, setf, de ciclos, de funções destrutivas ou de quaisquer funções com efeitos laterais é fortemente desincentivada dado que denota normalmente uma baixa qualidade técnica. Exceptua–se a utilização de setf em conjugação com gethash.

ATENÇÃO: Suspeitas confirmadas de plágio serão penalizadas com a anulação de ambos os projetos envolvidos (fonte e destino), e os responsáveis ficam sujeitos à instauração de processo disciplinar