Relatório Projeto LI2 - 'Reversi' - 2018/2019

Instituição: Universidade do Minho

UC: Laboratórios de Informática II

Alunos: João Pedro da Santa Guedes A89588

José Pedro Castro Ferreira A89572

Luís Pedro Oliveira de Castro Ferreira A89601

->Início do Projeto

Aquando da entrega deste projeto, após realizados os diversos TPCs propostos, rapidamente nos apercebemos que aquilo que tinhamos feito iria ser uma mais valia.

O projeto desafiou-nos a criar uma réplica do jogo 'Reversi' ou 'Othelio' com base em sucessivas impressões do tabuleiro no terminal, sendo que neste jogo deveriamos

ser capazes de jogar com 2 jogadores, alternadamente, ou contra um 'bot'.

->Desenvolvimento do Projeto

Conforme pedido no enunciado, o jogo deveria possuir comandos que correspondessem a: criar um novo jogo ('N'), guardar o jogo ('E'), ler um ficheiro de jogo previamente guardado ('L'),

começar um jogo contra um 'bot' ('A'), efetuar uma jogada ('J'), retroceder numa jogada ('U'), sair do jogo ('Q'), mostrar todas as jogadas válidas ('S') e assinalar a jogada mais benéfica ao utilizador ('H').

Para além disso existe o comando '?' que imprime todos os comandos disponíveis e a que corresponde a sua utilização e o comando 'P' que imprime o tabuleiro atual.

->Jogadas

Para ser possível jogar , além de ser necessário haver um jogo criado, existem algumas restrições às quais tivemos de obedecer: uma jogada só é válida se for possível cercar uma ou mais peças do oponente, em qualquer direção (utilização da função

'verificajogada') e as posições passíveis de serem jogadas são colocadas num tabuleiro auxiliar com jogadas válidas (função 'verificajogada2').

Quando se efetua uma jogada existe uma função (função 'joga') que transforma todas as peças cercadas nas peças contrárias.

->Sugestões de Jogada

Quando requesitado pelo jogador é possível indicarem-se quer todas as jogadas válidas quer a melhor jogada de acordo com o tabuleiro atual. Para mostrar todas as jogadas disponíveis é utilizada

a função 'whereCanIPut' que percorre o tabuleiro auxiliar com as jogadas válidas e nos imprime um '.' nesses locais. Já para mostrar a melhor jogada é utilizada a função 'giveHint' que utilizando um

algoritmo de alpha-beta pruning calcula a melhor jogada de acordo com o tabuleiro que nos é apresentado, e imprime um '?' nesse local.

->Condições de jogo

O jogo continua enquanto houver jogadas disponíveis por parte de um dos jogadores e enquanto o tabuleiro não se encontrar totalmente preenchido.

Para este efeito são utilizadas as funções 'gameOver', 'playerPlayable' e 'oppPlayable'.

->Ler/Gravar um jogo

Para gravar um jogo é criado um ficheiro ao qual o jogador dá o nome que quiser, e é nele gravado o estado atual do tabuleiro. Esta ação é realizada através da função 'guardajogo'.

Para ler um jogo, o jogador deve introduzir o nome do ficheiro guardado e a função 'readgame' vai interpretar o seu conteúdo.

->Voltar atrás nas jogadas

Também pedido no enunciado existe uma opção de retroceder quantas jogadas quisermos até ao início do jogo. Isto é possível através de estruturas dinâmicas, listas ligadas, em que existe um estado auxiliar que grava a jogada anterior, sendo assim possível voltar atrás.

->Estratégia do 'bot'

Inicialmente para o nosso bot começámos por utilizar o algoritmo de MinMax onde criámos um tabuleiro auxiliar preenchido com uma pontuação diferente para as variadas posições: aquelas que considerámos

mais valiosas para jogadas futuras valem mais que outras, e aquelas que podem prejudicar o 'bot' têm valores negativos.

No entanto após alguns testes notámos que a performance poderia ser melhorada e implementámos o algoritmo de alpha-beta pruming, que consiste também no cálculo da melhor jogada mas em vez de avaliar todas

as jogadas estipuladas, quando encontrar uma jogada que seja pior que a jogada avaliada anteriormente para e devolve a jogada anterior. Em comparação ao algoritmo de MinMax, o alpha-beta pruning reduz o número de jogadas avaliadas, diminuindo assim o tempo de resposta e aumentando a performance do nosso 'bot'.

Quanto aos diferentes níveis, a diferença no 'bot' é somente a quantidade de jogadas avaliadas, isto é, na função 'maxplay' o número de jogadas que ele vai avaliar é diferente. Para o nívle 1 o bot avalia 3 até jogadas, para o nível 2 avalia até 5 jogadas e para o nível 3 avalia até 7 jogadas. Como se torna evidente, quanto maior for o número possível de jogadas a avaliar, mais experiente será o 'bot' a prever qual a melhor jogada face ao tabuleiro atual e futuros tabuleiros que variam de acordo com todas as jogadas possíveis.

Dentro da função 'maxplay' ela averigua a cada possível jogada o número de pontos possíveis para o jogador e para o adversário, sendo que através da função 'contaPontos' vai calcular o valor máximo possível para o jogador e o mínimo para o adversário, com auxílio do tabuleiro que possui as cotações. Quando encontrar uma jogada que seja pior do que a calculada anteriormente, a função atualiza a linha e a coluna onde o 'bot' deve jogar e procede com a jogada do mesmo nessa posição.

->Conclusão

Face ao proposto, o grupo acha que conseguiu concretizar os desafios e criou uma réplica de 'Reversi', ou 'Othelio', jogável e com uma qualidade decente.

Talvez um dos aspetos a melhorar seja a performance do 'bot' e fazer uma maior distinção a nível das estratégias para os diferentes níveis.