

### Aprendizagem por Reforço em Jogo do Tipo Solitário

Relatório Final

Inteligência Artificial

3º ano do Mestrado Integrado em Engenharia Informática e Computação

Elementos do Grupo:

Diogo Mendes – 201605360

Diogo Sousa – 201706409

28 de maio de 2020

# Índice

1. Objetivo
2. Especificação
   1. Análise do tema
   2. Ilustração de cenários
   3. Abordagem
      1. Q-Learning
      2. SARSA
3. Desenvolvimento
   1. Estrutura da Aplicação
4. Experiências
5. Conclusões
6. Melhoramentos
7. Recursos
   1. Bibliografia
   2. Software
8. Apêndice
   1. Manual de Utilizador

# Objetivo

Este projeto foi desenvolvido no âmbito da cadeira de Inteligência Artificial e tem como objetivo a aplicação de algoritmos de aprendizagem por reforço num jogo do tipo solitário, neste caso o BoxWorld 2. Pretende-se, com isto, treinar um agente de forma a resolver os níveis do jogo no menor número de movimentos possíveis.

# Especificação

## Análise do tema

Como mencionado anteriormente, este projeto consiste na aplicação de algoritmos de aprendizagem por reforço num jogo do tipo solitário, neste caso o BoxWorld 2, no contexto do problema.

O BoxWorld 2 consiste num jogo de puzzle em que o jogador controla uma personagem num mundo 2D visto de cima com o objetivo de chegar ao fim de cada nível. No entanto para fazer isso tem de movimentar certas caixas para poder ter um caminho a percorrer. A personagem pode empurrar as caixas para a posição oposta da sua posição desde que o espaço esteja vazio. O objetivo é completar cada nível com o mínimo de movimentos e ‘empurrões’ possível.

## Ilustração de cenários

Decidimos criar sete níveis do jogo, na forma de matrizes 8 por 8, onde cada matriz corresponde a um array 2D. Cada posição no array representa algo diferente: o jogador (‘J’), a saída do nível (‘S’), uma caixa (‘C’), uma parede (‘P’) ou um espaço vazio (‘\_’). Formulamos quatro funções de movimento (Para mover o jogador para cima, baixo, esquerda ou direita) que são usadas para explorar uma possível solução. Cada movimento tem como objetivo resultar na solução (quando o Jogador se move para a Saída) ou quando não é possível, para um array que não tenha sido representado ainda (para evitar loops infinitos). Para além disso, o programa tenta chegar à solução com o mínimo de movimentos possível, visto que cada ‘passo’ do programa também é um passo do jogo, isto torna avaliar a qualidade de uma solução fácil.

## Abordagem

Perante o problema, foi decidida a utilização de Algoritmos de Aprendizagem por Reforço, nomeadamente o Q-Learning e o SARSA.

### Q-Learning

Nesta primeira fase, é necessário definir um indivíduo pertencente à população. Um indivíduo é tipicamente representado em forma binária, onde cada 0 e 1 é nomeado de alelo. Esta representação permite definir um indivíduo como um cromossoma.

Q-learning trata-se de um algoritmo de aprendizagem por reforço cujo objetivo é ensinar um agente a decidir que decisão tomar dentro de determinadas circunstâncias. Não requere um modelo do ambiente, e consegue lidar com variados problemas, sem qualquer tipo de adaptação.

Este algoritmo encontra uma política ótima, no sentido que maximiza o valor esperado do total da recompensa dentro de todos os vários passos sucessivos, começando pelo estado atual.

### SARSA

A definição deste algoritmo encontra-se escondida no seu nome, ou seja, a principal função para atualizar o Q-value depende do estado atual do agente “S1”, da ação que o agente escolhe tomar “A1”, da recompensa que o agente recebe por escolher essa ação, do estado “S2” em que o agente entra depois de tomar essa mesma ação e, finalmente, da próxima ação que o agente escolha tomar “A2” no seu novo estado.

Um agente SARSA interage com o ambiente e atualiza a política baseada nas ações tomadas. O Q-value de um estado-ação é atualizado por um erro, ajustado pela taxa de aprendizagem *alpha*. Os Q-values representam a possível recompensa recebida por tomar uma determinada ação num determinado estado, e ainda desconta uma futura recompensa recebida pela próxima decisão que tenha que tomar no estado seguinte.

Algumas otimizações do Q-Learning podem ser aplicadas ao SARSA.

# Desenvolvimento

O programa foi desenvolvido em C++, utilizando o Eclipse como IDE, no ambiente de desenvolvimento Windows.

## Estrutura da Aplicação

A aplicação foi elaborada em três ficheiros cpp, dois contendo cada algoritmo de reforço utilizado e outro contendo diversas funções, como as quatro funções de movimento do jogador, funções de teste, menu, regras do jogo, entre outras.

Por fim, a interface gráfica é simples, expondo claramente cada nível para facilitar a interação do utilizador com o programa, bem como a introdução de dados por parte de quem utiliza o programa.

# Experiências

Resultados

# Conclusões

O desenvolvimento do projeto foi, em grande parte, despendido na implementação dos algoritmos visto que já tínhamos implementado a estrutura do jogo no primeiro trabalho.

A utilização de algoritmos de aprendizagem por reforço fornece um ganho significativo no que diz respeito ao tempo gasto para se solucionar o problema, pois é possível, em apenas algumas iterações, chegar rapidamente a soluções válidas, formando rapidamente uma solução ideal.

Por conseguinte, os algoritmos de aprendizagem por reforço ajudam sempre a chegar a uma solução cada vez melhor. Num ponto negativo, a solução que obtemos pode não ser a solução ótima, mas não deixa de ser uma boa aproximação.

# Melhoramentos

A introdução de outros algoritmos de aprendizagem por reforço como PPO ou SAC poderiam vir a tornar o nosso programa mais completo e com soluções ainda mais precisas.

Uma eventual utilização do Unity e ML agentes também seria outra possível melhoria.

# Recursos

## Bibliografia

* + 1. <https://www.cse.unsw.edu.au/~cs9417ml/RL1/algorithms.html>
    2. <https://github.com/rhiever/Data-Analysis-and-Machine-Learning-Projects/blob/master/example-data-science-notebook/Example%20Machine%20Learning%20Notebook.ipynb>

## Software

1. Eclipse IDE for C++
2. Windows
3. Github

# Apêndice

## Manual de Utilizador

Para correr o programa, basta importar o projeto para Eclipse e correr a partir deste último.

No início da execução, será apresentado um menu sobre como executar o programa. Após o preenchimento de certos valores, o programa correrá o algoritmo escolhido e, após terminar, será mostrada a melhor solução.