**Algoritmo Q-Learning no ambiente Agent0**

Beatriz Silva (2019108992@uac.pt), Pedro Sousa (2019101451@uac.pt), Salif Faustino ([20172005@uac.pt](mailto:20172005@uac.pt))

**Resumo**

Este trabalho foi realizado no âmbito da disciplina de Inteligência Artificial e teve como objetivo a exploração e interação entre um agente e um ambiente. O software consiste num ambiente onde o agente se consegue movimentar. Com base nas experiências efetuadas com o algoritmo escolhido, foi possível interpretar e perceber os diversos cenários a que o agente foi submetido.

**Introdução**

Na cadeira de Inteligência Artificial, do 3º Ano da Licenciatura em Informática, foi-nos proposto a interpretação e implementação de um algoritmo no Agente0, sendo assim, decidimos utilizar algoritmo **Q-Learning.** A versão utilizada neste projeto foi o Agent0\_Minotauro\_Reinforcement, que permite explorar a interação entre um agente e um ambiente. Este software consiste num ambiente com um tabuleiro retangular de casas quadradas onde o agente se pode movimentar. De modo a movimentar-se, o agente pode deslocar-se em frente ou mudar de direção e dependendo da sua capacidade, pode também inspecionar o tabuleiro e as suas casas. O agente pode utilizar diversos algoritmos para chegar ao seu objetivo e é possível observar as casas com diversas cores. A interação entre o agente e o ambiente é comandada através de um cliente e acontece no servidor. A versão final do programa pode ser encontrada no github, através do link seguinte:

<https://github.com/SalifNTC/projeto-2-g10>

Link do vídeo:

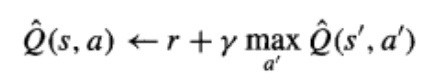
<https://youtu.be/sTyNQGC9T6M>

**Descrição do Algoritmo**

O algoritmo **Q-Learning** é um algoritmo de aprendizagem por reforço, com uma política que procura encontrar a melhor ação a ser executada no estado atual. O **Q-Learning** é um algoritmo off-policy, podendo assim atualizar as funções de valor estimado usando ações que ainda não foram experimentadas. O **Q-Learning** aprende a política ideal, mesmo quando as ações são selecionadas de acordo com uma política mais exploratória ou até aleatória.

O **“Q”** em que **Q-Learning** representa o quão útil uma determinada ação é para ganhar alguma recompensa.

Quando o algoritmo é utilizado é criada uma tabela designada por **Q-table** que obedece o seguinte formato: [estado, ação]. Os valores são inicializados a zero. A atualização desses valores-q sempre que ocorre um episódio. Esta tabela torna-se uma referência para o agente selecionar a melhor ação com base no valor q.



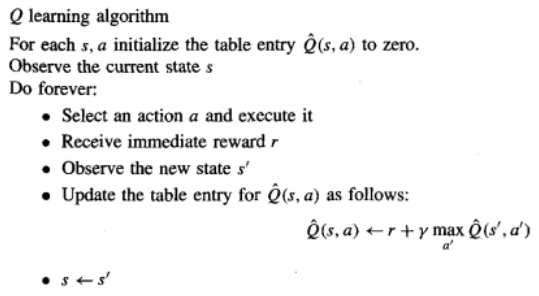
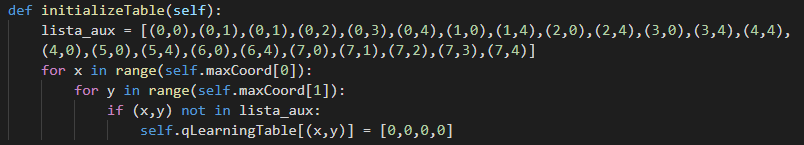


Figura 1- Algoritmo Q-Learning

**Implementação do Q-Learning**

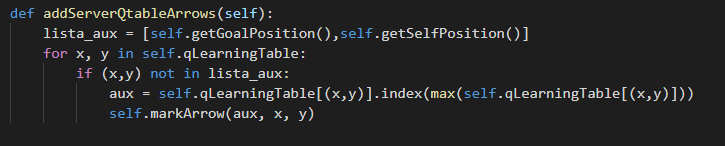
A tabela ***(Figura 2) Q – table*** é iniciada com valores a zero. Foi criada uma lista auxiliar adaptada ao mapa utilizado com as localizações dos obstáculos, posição inicial, e objetivo. Esta lista é alterada quando modificamos o mapa. Isto deveu-se à falta de tempo e experiência para atualizar esta lista com as localizações dos obstáculos, sendo assim feita de maneira rudimentar no código.

Figura 2 – Inicialização da Q-Table

Atualização ***(Figura 3)*** da tabela utilizado a fórmula reward = reward(local anterior) + 0.9\*reward(local). Isto ocorre depois do agente percorrer o caminho aleatoriamente.

Figura 3 – Atualização da Q-Table

Função utilizada ***(Figura 4)*** para criar as setas no mapa, baseadas no valor das recompensas na tabela. As setas apontam no sentido do valor máximo das recompensas.

Figura 4 – Adição de setas ao mapa

Na ***Figura 5*** o agente inicia a tabela da ***Figura 2*,** a posição do objetivo e o número de interações a executar (Neste caso, 100). Aqui, este executa ações aleatórias até atingir o objetivo, onde faz update da tabela **(*Figura 3*)**. No final é apresentada a tabela final e apresentadas as setas que indicam o caminho a seguir para o objetivo em cada ponto explorado ***(Figura 4*)**.

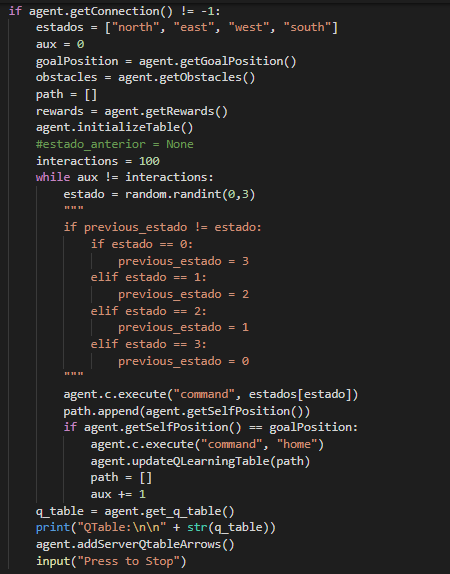


Figura 5 – Execução e pesquisa realizada pelo agente

**Experiências Realizadas ou Exemplos de Aplicação**

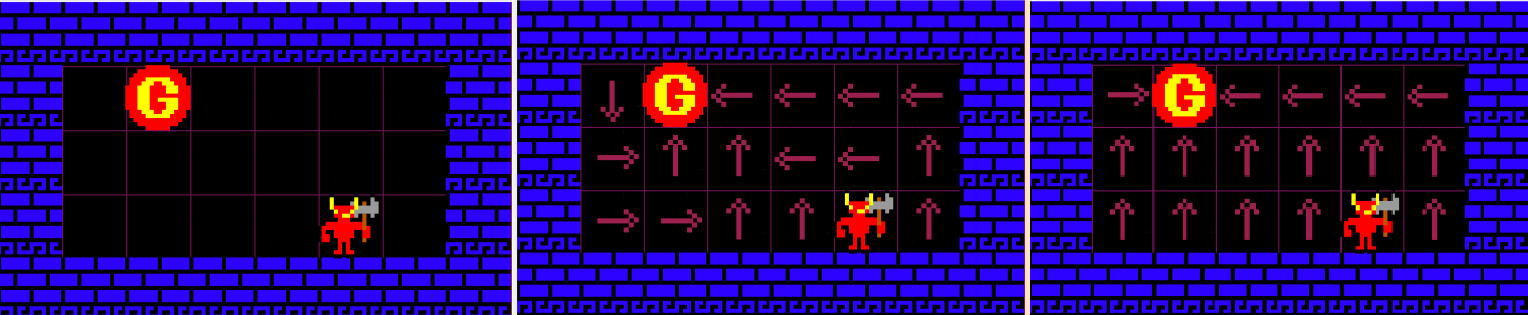
No conjunto de experiências apresentadas consideramos três estados fundamentais:

* Agente no ambiente;
* Agente após execução do algoritmo com 5 iterações;
* Agente após execução do algoritmo com 50 iterações**.**

**Experiência 1, mundo sem obstáculos/target:**

Nesta experiência ***(figura 6)***, o agente encontra-se inserido num mundo sem obstáculos/ target, o que faz com que, com a implementação do algoritmo **Q-learning,** o agente percorra o mapa sem ter em conta os mesmos.

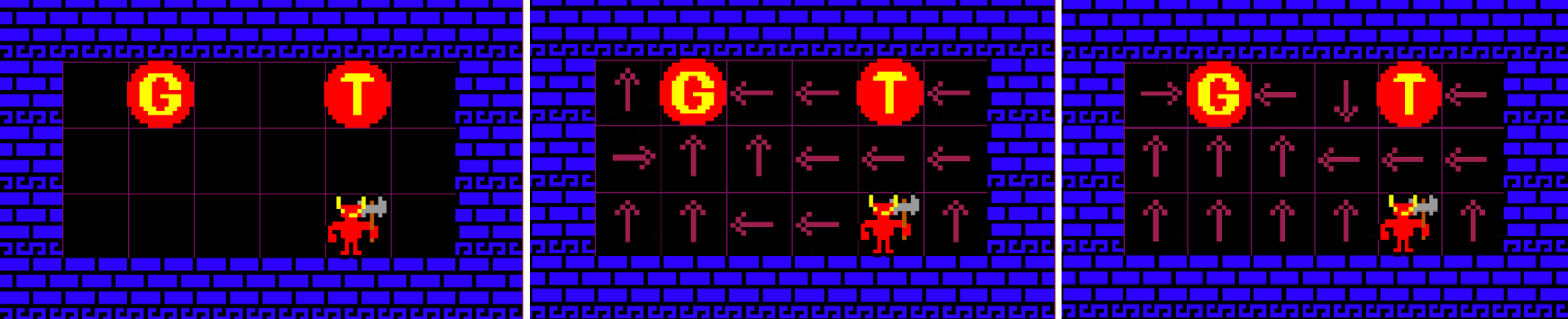
Com base nas imagens da ***figura 6***, é possível perceber que os “caminhos” indicados após a implementação do algoritmo variam consoante ao número de iterações, ou seja, se as interações realizadas forem poucas, o agente pode não ter dados relativos a determinadas casas. Tal como se pode verificar, na imagem central, com poucas interações o agente não denota o caminho que deveria fazer na casa no canto superior esquerdo seria diretamente para o objetivo. Isto deve-se, provavelmente, ao agente não ter percorrido essa casa. No entanto, na imagem mais à direita, o agente já identifica propriamente o caminho a executar.

*Figura 6 –* Mundo sem obstáculos / Target

**Experiência 2, mundo com obstáculos/target:**

Nesta experiência **(figura 7)**, o agente encontra-se inserido num mundo com obstáculos/target.

A grande particularidade nesta figura foi a utilização do target, onde é possível perceber que com a ocorrência dos episódios o algoritmo não coloca a seta na posição onde encontra-se o target (imagem central, imagem mais a direita).

*Figura 7* – Mundo com obstáculos / Target

**Discussão e Conclusão**

Acreditamos ter conseguido alcançar o nosso objetivo na realização do projeto, obtendo uma boa simulação para encontrar o objetivo. O agente atualiza os valores da **Q-Table** ao fim de cada episódio, resultando numa tabela que produz resultados esperados ao fim de muitos episódios.

Para um pequeno número de episódios, o agente é capaz de cometer ligeiros erros nos caminhos mais eficientes, sendo que, por isso, é aconselhável ter um maior número de episódios.

No entanto, devido à política utilizada ser aleatória, é possível haver uma divergência de resultados para mundos maiores ou com a existência de diversos objetivos.

**Bibliografia**

Slides fornecidos pelo professor José Cascalho.

Livro Machine Learning - Tom Mitchell