Algoritmo de Reforço no Agent0-RL

André Sousa (20182451@uac.pt), Bruno Viveiros (20182149@uac.pt), Gonçalo Almeida (20182448@uac.pt)

Resumo

O objetivo deste projeto foi o implementar o algoritmo Q-Learning adaptado, inserido no contexto da Aprendizagem por Reforço. O Agente é colocado num ambiente simples, onde é colocada uma casa objetivo (G) e em alguns cenários um alvo (T). Dependendo do número de episódios, e à medida que o número destes é incrementado, o agente consegue melhor preencher a matriz Q-learning com as políticas, através do cálculo da recompensa, estado a estado, sendo que por fim, e no caso de serem utilizados episódios suficientes, estas apresentam os resultados a convergir para a casa objetivo (G).

Introdução

Para o projeto, foi utilizado o código do Agent0_minotauro_RL, fornecido pelo professor. Este consiste num "mapa", e num "agente" que possui diversas funcionalidades de interação com o ambiente, explorando o mundo de forma aleatória. O objetivo foi o de testar o conceito de um algoritmo de Aprendizagem por Reforço, o Q-Learning adaptado, sendo que este só atualiza os valores de recompensa no final de cada episódio, mostrando as políticas selecionadas pelo agente após ter explorado o mundo um certo número de vezes.

GitHub
https://github.com/brun9/Agent0_Reinforced_Learning

Vídeo
https://www.youtube.com/watch?v=vyc43jbBnO4

Descrição do Algoritmo

O algoritmo utilizado foi o Q-learning adaptado, em que o agente através de uma exploração aleatória do mundo contabilizada em vários episódios, vai calculando o melhor comportamento a ter para cada um dos estados do mapa. Cada episódio termina quando o agente atinge a casa objetivo (G) ou a casa alvo (T). Ao fim de um determinado número de episódios, é alterada a matriz Q-learning, com as recompensas para cada estado e ação, sendo então apresentadas as melhores escolhas do agente, denonimadas de "políticas". Estas são representativas do comportamento ótimo do agente, em cada posição do tabuleiro e baseadas no cálculo da recompensa para cada ação, em relação a cada estado.

De modo a efetuar os cálculos, o algoritmo usa a fórmula:

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s',a')$$

O cálculo de Q é efetuado através da soma de cada recompensa "r" com a multiplicação entre a parâmetro de desconto "y", cujo valor utilizado foi de 0,9 e a maior recompensa para cada estado e ação. O agente desloca-se aleatoriamente até encontrar o objetivo, ou o target, e quando o faz, ajustam-se os valores da matriz Q-learning, ao fim de cada episódio. Quando terminados os episódios, são então representadas, com setas, as políticas finais calculadas pelo algoritmo, baseadas nos valores de maior recompensa presente na matriz. Caso haja empate de valores, a prioridade de escolha definida foi Norte, Este, Sul, Oeste.

Experiências Realizadas

Foram realizadas três experiências distintas, de modo a avaliar a diferença no cálculo das políticas para diversas situações apresentadas. Em cada estado são então dispostas as setas representativas da política em função da melhor ação a tomar para cada estado. Para as três experiências, foi utilizado um ambiente com dimensões 3x7, variando então a existência ou não de uma casa alvo (T) e o número de episódios.

Tabela de recompensas:

Para as nossas experiências, o valores das recompensas utilizadas foram:

Casa Normal	Casa Objetivo	Casa Alvo
0	100	-50

Experiência 1

Na seguinte experiência, tanto o objetivo como a casa de partida estão nas posições originais (Fig 1a). Foram realizados 50 episódios, os quais foram concluídos em aproximadamente 6 minutos. Os resultados obtidos (Fig 1b) demonstram uma convergência das políticas para a casa objetivo, indicando que o algoritmo funcionou como esperado.

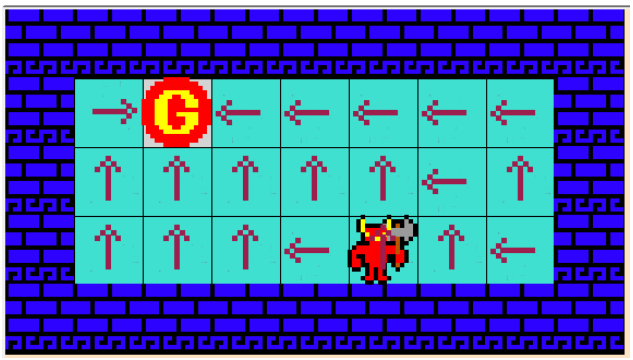
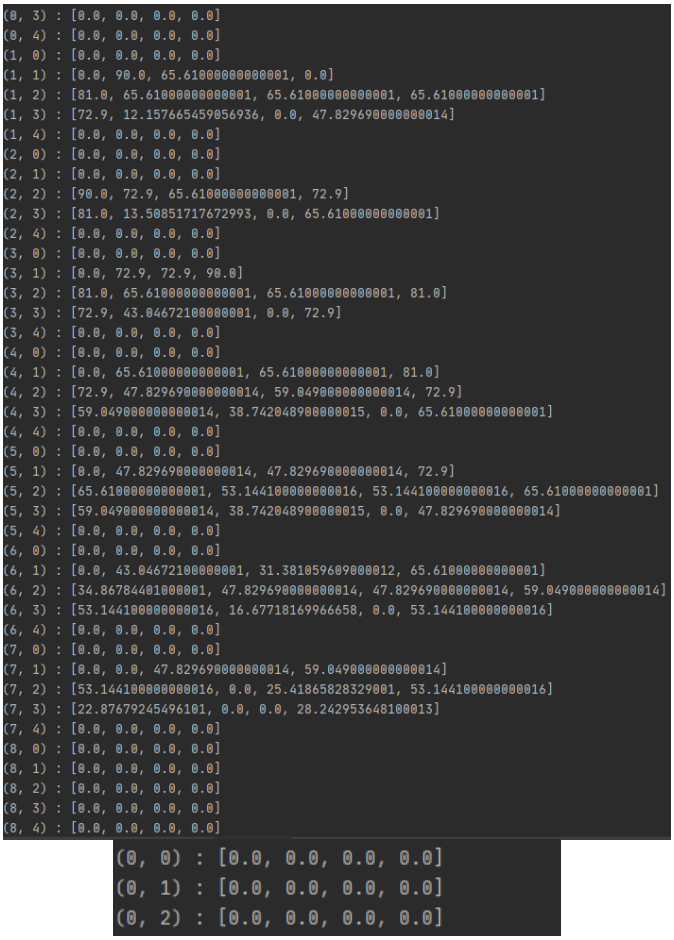


Fig 1a: Cenário em não existe qualquer alvo.



```
(0, 3) : [0.0, 0.0, 0.0, 0.0]
(0, 4) : [0.0, 0.0, 0.0, 0.0]
(1, 0) : [0.0, 0.0, 0.0, 0.0]
(1, 1) : [0.0, 90.0, 65.61000000000001, 0.0]
(1, 2) : [81.0, 65.61000000000001, 65.61000000000001, 65.61000000000001]
(1, 3) : [72.9, 12.157665459056936, 0.0, 47.829690000000014]
(1, 4) : [0.0, 0.0, 0.0, 0.0]
(2, 0) : [0.0, 0.0, 0.0, 0.0]
(2, 1) : [0.0, 0.0, 0.0, 0.0]
(2, 2) : [90.0, 72.9, 65.61000000000001, 72.9]
(2, 3) : [81.0, 13.50851717672993, 0.0, 65.61000000000001]
(2, 4) : [0.0, 0.0, 0.0, 0.0]
(3, 0) : [0.0, 0.0, 0.0, 0.0]
(3, 1) : [0.0, 72.9, 72.9, 90.0]
(3, 2) : [81.0, 65.61000000000001, 65.61000000000001, 81.0]
(3, 3) : [72.9, 43.046721000000001, 0.0, 72.9]
(3, 4) : [0.0, 0.0, 0.0, 0.0]
(4, 0) : [0.0, 0.0, 0.0, 0.0]
(4, 1) : [0.0, 65.61000000000001, 65.61000000000001, 81.0]
(4, 2) : [72.9, 47.829690000000014, 59.049000000000014, 72.9]
(4, 3) : [59.049000000000014, 38.742048900000015, 0.0, 65.61000000000001]
(4, 4) : [0.0, 0.0, 0.0, 0.0]
(5, 0) : [0.0, 0.0, 0.0, 0.0]
(5, 1) : [0.0, 47.829690000000014, 47.829690000000014, 72.9]
(5, 2) : [65.61000000000001, 53.144100000000016, 53.144100000000016, 65.61000000000001]
(5, 3) : [59.049000000000014, 38.742048900000015, 0.0, 47.829690000000014]
(5, 4) : [0.0, 0.0, 0.0, 0.0]
(6, 0) : [0.0, 0.0, 0.0, 0.0]
(6, 1) : [0.0, 43.046721000000001, 31.381059609000012, 65.61000000000001]
(6, 2) : [34.86784401000001, 47.829690000000014, 47.829690000000014, 59.049000000000014]
(6, 3) : [53.144100000000016, 16.677181699666658, 0.0, 53.144100000000016]
(6, 4) : [0.0, 0.0, 0.0, 0.0]
(7, 0) : [0.0, 0.0, 0.0, 0.0]
(7, 1) : [0.0, 0.0, 47.829690000000014, 59.049000000000014]
(7, 2) : [53.144100000000016, 0.0, 25.41865828329001, 53.144100000000016]
(7, 3) : [22.87679245496101, 0.0, 0.0, 28.242953648100013]
(7, 4) : [0.0, 0.0, 0.0, 0.0]
(8, 0) : [0.0, 0.0, 0.0, 0.0]
(8, 1) : [0.0, 0.0, 0.0, 0.0]
(8, 2) : [0.0, 0.0, 0.0, 0.0]
(8, 3) : [0.0, 0.0, 0.0, 0.0]
(8, 4) : [0.0, 0.0, 0.0, 0.0]
(0, 0) : [0.0, 0.0, 0.0, 0.0]
(0, 1) : [0.0, 0.0, 0.0, 0.0]
(0, 2) : [0.0, 0.0, 0.0, 0.0]
```

Fig 1b: Valores da matriz Q-learning, Experiência 1.

Experiência 2

Na seguinte experiência, foi utilizado o mesmo mapa que na anterior, com a adição de uma casa alvo (Fig 2a), de modo a avaliar as diferenças de comportamento do algoritmo nas duas situações. Foram realizados 200 episódios, os quais foram concluídos após sensivelmente 16 minutos. Os resultados desta experiência (Fig 2b) mostram que na presença de um target, com recompesa -50, o algoritmo define as políticas de modo a evitar a passagem nessa casa, o que nos leva a concluír que o mesmo foi bem sucedido.

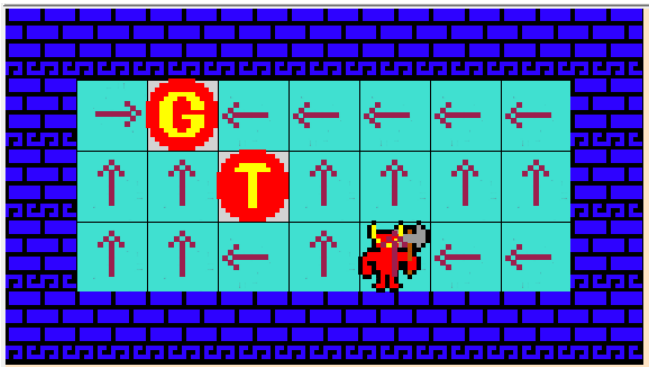


Fig 2a: Cenário em que, para além da casa objetivo (G) existe uma casa alvo (T) onde o agente também termina a sua tentativa.

```
(0, 3) : [0.0, 0.0, 0.0, 0.0]
(0, 4) : [0.0, 0.0, 0.0, 0.0]
(1, 0) : [0.0, 0.0, 0.0, 0.0]
(1, 1) : [0.0, 90.0, 28.242953648100013, 0.0]
(1, 2) : [81.0, 0.0, 59.049000000000014, 0.0]
(1, 3) : [72.9, 0.02184745005283925, 0.0, 0.0]
(1, 4) : [0.0, 0.0, 0.0, 0.0]
(2, 0) : [0.0, 0.0, 0.0, 0.0]
(2, 1) : [0.0, 0.0, 0.0, 0.0]
(2, 2) : [90.0, 0.0, 65.61000000000001, 47.829690000000014]
(2, 3) : [81.0, 0.024274944503154722, 0.0, 65.61000000000001]
(2, 4) : [0.0, 0.0, 0.0, 0.0]
(3, 0) : [0.0, 0.0, 0.0, 0.0]
(3, 1) : [0.0, 53.144100000000016, 0.0, 90.0]
(3, 2) : [0.0, 0.0, 0.0, 0.0]
(3, 3) : [0.0, 59.049000000000014, 0.0, 72.9]
(3, 4) : [0.0, 0.0, 0.0, 0.0]
(4, 0) : [0.0, 0.0, 0.0, 0.0]
(4, 1) : [0.0, 59.049000000000014, 59.049000000000014, 81.0]
(4, 2) : [72.9, 59.049000000000014, 28.242953648100013, 0.0]
(4, 3) : [65.61000000000001, 43.04672100000001, 0.0, 65.61000000000001]
(4, 4) : [0.0, 0.0, 0.0, 0.0]
(5, 0) : [0.0, 0.0, 0.0, 0.0]
(5, 1) : [0.0, 34.86784401000001, 38.742048900000015, 72.9]
(5, 2) : [65.61000000000001, 47.829690000000014, 43.04672100000001, 65.61000000000001]
(5, 3) : [59.049000000000014, 43.04672100000001, 0.0, 59.049000000000014]
(5, 4) : [0.0, 0.0, 0.0, 0.0]
(6, 0) : [0.0, 0.0, 0.0, 0.0]
(6, 1) : [0.0, 53.144100000000016, 47.829690000000014, 65.61000000000001]
(6, 2) : [53.144100000000016, 28.58911320946491, 47.829690000000014, 53.144100000000016]
(6, 3) : [43.04672100000001, 34.86784401000001, 0.0, 53.144100000000016]
(6, 4) : [0.0, 0.0, 0.0, 0.0]
(7, 0) : [0.0, 0.0, 0.0, 0.0]
(7, 1) : [0.0, 0.0, 34.86784401000001, 59.049000000000014]
(7, 2) : [47.829690000000014, 0.0, 43.04672100000001, 38.742048900000015]
(7, 3) : [43.04672100000001, 0.0, 0.0, 47.829690000000014]
(7, 4) : [0.0, 0.0, 0.0, 0.0]
(8, 0) : [0.0, 0.0, 0.0, 0.0]
(8, 1) : [0.0, 0.0, 0.0, 0.0]
(8, 2) : [0.0, 0.0, 0.0, 0.0]
(8, 3) : [0.0, 0.0, 0.0, 0.0]
(8, 4) : [0.0, 0.0, 0.0, 0.0]
```

```
(0, 0) : [0.0, 0.0, 0.0, 0.0]
(0, 1) : [0.0, 0.0, 0.0, 0.0]
(0, 2) : [0.0, 0.0, 0.0, 0.0]
(0, 3) : [0.0, 0.0, 0.0, 0.0]
```

Fig 2b: Valores da matriz Q-learning, Experiência 2.

Experiência 3

Na seguinte experiência a casa objetivo e casa alvo utilizadas anteriormente mantêm-se nas mesmas posições. Para além disso, foi adicionada uma casa alvo nova (Fig 3a), de modo a analisar a diferença dos resultados obtidos. Os resultados desta experiência (Fig 3b) mostram que mesmo na presença de dois targets, o algoritmo define as políticas de modo a evitar a passagem nessas casas, o que nos leva a concluír que o mesmo continua a ser bem sucedido nesta situação. Foram realizados 750 episódios, com uma duração total de sensivelmente uma hora.

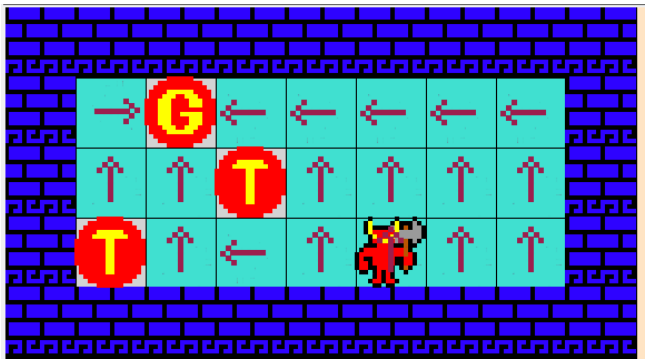


Fig 3a: Cenário em que, para além da casa objetivo (G) e a já existente casa alvo (T), é adicionada mais uma casa alvo (T).

```
(0, 3) : [0.0, 0.0, 0.0, 0.0]
(0, 4) : [0.0, 0.0, 0.0, 0.0]
(1, 0) : [0.0, 0.0, 0.0, 0.0]
(1, 1) : [0.0, 90.0, 0.0, 0.0]
(1, 2) : [81.0, 0.0, 0.0, 53.144100000000016]
(1, 3) : [0.0, 0.0, 0.0, 0.0]
(1, 4) : [0.0, 0.0, 0.0, 0.0]
(2, 0) : [0.0, 0.0, 0.0, 0.0]
(2, 1) : [0.0, 0.0, 0.0, 0.0]
(2, 2) : [90.0, 0.0, 72.9, 72.9]
(2, 3) : [81.0, 38.742048900000015, 0.0, 65.61000000000001]
(2, 4) : [0.0, 0.0, 0.0, 0.0]
(3, 0) : [0.0, 0.0, 0.0, 0.0]
(3, 1) : [0.0, 72.9, 0.0, 90.0]
(3, 2) : [0.0, 0.0, 0.0, 0.0]
(3, 3) : [0.0, 53.144100000000016, 0.0, 72.9]
(3, 4) : [0.0, 0.0, 0.0, 0.0]
(4, 0) : [0.0, 0.0, 0.0, 0.0]
(4, 1) : [0.0, 65.61000000000001, 65.61000000000001, 81.0]
(4, 2) : [72.9, 59.049000000000014, 59.049000000000014, 0.0]
(4, 3) : [65.61000000000001, 53.144100000000016, 0.0, 65.61000000000001]
(4, 4) : [0.0, 0.0, 0.0, 0.0]
(5, 0) : [0.0, 0.0, 0.0, 0.0]
(5, 1) : [0.0, 59.049000000000014, 59.049000000000014, 72.9]
(5, 2) : [65.61000000000001, 47.829690000000014, 47.829690000000014, 65.61000000000001]
(5, 3) : [59.049000000000014, 47.829690000000014, 0.0, 59.049000000000014]
(5, 4) : [0.0, 0.0, 0.0, 0.0]
(6, 0) : [0.0, 0.0, 0.0, 0.0]
(6, 1) : [0.0, 53.144100000000016, 47.829690000000014, 65.61000000000001]
(6, 2) : [59.049000000000014, 38.742048900000015, 43.04672100000001, 59.049000000000014]
(6, 3) : [53.144100000000016, 38.742048900000015, 0.0, 47.829690000000014]
(6, 4) : [0.0, 0.0, 0.0, 0.0]
(7, 0) : [0.0, 0.0, 0.0, 0.0]
(7, 1) : [0.0, 0.0, 43.04672100000001, 59.049000000000014]
(7, 2) : [53.144100000000016, 0.0, 38.742048900000015, 53.144100000000016]
(7, 3) : [47.829690000000014, 0.0, 0.0, 43.04672100000001]
(7, 4) : [0.0, 0.0, 0.0, 0.0]
(8, 0) : [0.0, 0.0, 0.0, 0.0]
(8, 1) : [0.0, 0.0, 0.0, 0.0]
(8, 2) : [0.0, 0.0, 0.0, 0.0]
(8, 3) : [0.0, 0.0, 0.0, 0.0]
(8, 4) : [0.0, 0.0, 0.0, 0.0]
(0, 0) : [0.0, 0.0, 0.0, 0.0]
(0, 1) : [0.0, 0.0, 0.0, 0.0]
(0, 2) : [0.0, 0.0, 0.0, 0.0]
```

Fig 3b: Valores da matriz Q-learning, Experiência 3.

Discussão e Conclusão

Tendo em conta os resultados obtidos para as diversas experiências realizadas, foi observado que o algoritmo utilizado funciona consideravelmente bem para o cálculo de políticas num ambiente de pouca dimensão e complexidade. Isto não acontece para o caso de ambientes mais complexos, principalmente se a exploração é feita somente aleatoriamente, o que resulta num tempo de execução exagerado. Não obstante, o incremento do número de episódios realizados aparenta ter um impacto significativo nos resultados obtidos para estes casos, levando-nos à conclusão que quantos mais alvos colocados no ambiente, maior terá de ser o número de episódios de modo a que o algoritmo consiga apresentar resultados desejados.

Bibliografia

1. Russell, S. J., Norvig, P., Davis, E. (2010). *Artificial intelligence: a modern approach.* 3rd ed. Upper Saddle River, NJ: Prentice Hall.
2. Sutton, R. S., Barto, A. G. (2014-2015). Reinforcement Learning: An Introduction. 2nd ed. The MIT Press