UFMG

Trabalho Prático 2

Q-Learning

Arthur Pontes Nader - 2019022294

1) Introdução

O aprendizado por reforço é uma abordagem em que um agente interage com um ambiente, tomando ações e recebendo recompensas de acordo com seu desempenho. Assim, o agente utiliza essas recompensas para aprender uma política que maximize seus ganhos futuros.

Dessa maneira, o objetivo deste trabalho é realizar a implementação de um agente de aprendizado por reforço utilizando o algoritmo Q-Learning. Por meio desse algoritmo, o agente aprende uma tabela de valores Q, que representa a recompensa esperada para cada combinação possível de estado e ação existente.

2) Implementação

classe Agente

A classe "Agente" é responsável por implementar a lógica de um agente que realiza aprendizado por reforço em um ambiente. Ela possui os seguintes atributos e métodos:

Atributos:

- **tabela_q:** Uma matriz de dimensão (N, N, 4) que representa os valores Q, ou seja, as recompensas esperadas para cada combinação possível de estado (posição do ambiente) e ação.
- **ambiente:** matriz que representa o ambiente em que o agente está se movendo.
- **num_iteracoes:** número de iterações que o agente fará durante a exploração do ambiente.
- **taxa:** taxa de aprendizado que será utilizada para atualizar os valores da tabela Q.
- **desconto:** fator de desconto utilizado para ponderar as recompensas futuras na atualização dos valores da tabela Q.
- **recompensa:** valor da recompensa obtida pelo agente nas posições do ambiente que possuem valor igual a 0

- **eps:** O valor de epsilon utilizado para determinar a probabilidade de exploração ou explotação durante a tomada de decisões.
- **posicao_inicial:** guarda as coordenadas iniciais do agente no ambiente.
- **evolucao:** lista que guarda o estado do ambiente após cada iteração do algoritmo.

Métodos:

- **__init__(self, N, ambiente_, iteracoes, taxa_aprendizado, fator_desconto, recompensa_, epsilon):** esse é o método de inicialização da classe. Ele recebe os parâmetros lidos do arquivo para configurar o agente, o ambiente e definir os valores iniciais dos atributos.
- **executar_acao(self, acao, posicao):** Este método recebe uma ação e uma posição atual do agente e retorna a nova posição e a recompensa do estado resultante da ação.
- **explorar_ambiente(self):** Este método executa a exploração do ambiente pelo agente. Com base na tabela Q, toma decisões e atualiza os valores da tabela. Além disso, armazena a evolução do ambiente em cada iteração.
- **gerar_gif(self):** por meio do atributo “evolucao”, esse método gera um arquivo de animação GIF que mostra a evolução do ambiente durante a exploração.
- **gerar_imagem(self):** Este método gera uma imagem (heatmap) que representa as melhores ações em cada posição da matriz do ambiente, com base nos valores da tabela Q.

Além disso, há uma função chamada ler_arquivo(caminho_arquivo), que por meio da leitura dos dados do arquivo passado, gera os parâmetros para iniciar a execução. A execução é basicamente instanciar um objeto do tipo Agente com esses parâmetros lidos e em seguida chamar os métodos explorar_ambiente(), gerar_imagem() e gerar_gif().

3) Descrição do algoritmo Q-Learning

O algoritmo Q-Learning é uma técnica de aprendizado por reforço que permite a um agente aprender a tomar decisões em um ambiente desconhecido, visando maximizar suas recompensas ao longo do tempo. Ele utiliza uma tabela chamada de tabela Q, onde cada entrada representa o valor esperado das recompensas futuras para uma determinada combinação de estado e ação.

O algoritmo Q-Learning implementado possui os seguintes passos:

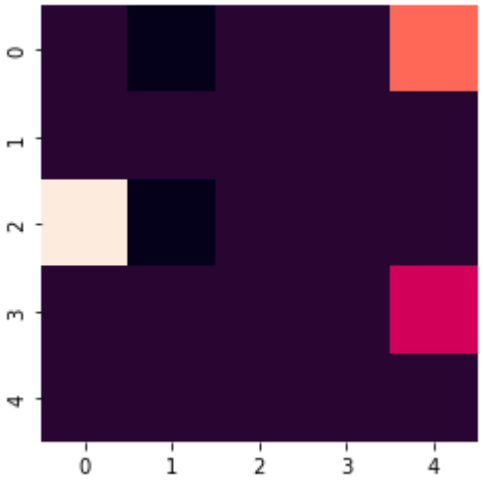
- Iniciar a tabela Q com valores 0 para todas as combinações possíveis de estado e ação.
- Primeiro, o agente seleciona uma ação para ser executada com base em uma estratégia de exploração e explotação, o que significa que pode escolher uma ação aleatória com uma certa probabilidade (exploração) ou selecionar a ação com o maior valor Q para o estado atual (explotação). O agente pode escorregar durante a explotação e ir para posições perpendiculares com probabilidade de 20% (10% para cada direção).
- Em seguida, o agente executa a ação escolhida e observa o próximo estado do ambiente, bem como a recompensa associada a essa transição de estado.
- Com base na recompensa observada e no próximo estado, o agente atualiza o valor da tabela Q para o par estado-ação correspondente. A seguinte fórmula é utilizada:

$$Q(s, a) = Q(s, a) + \text{taxa\_de\_aprendizado} * (\text{recompensa} + \text{fator\_de\_desconto} * \max[Q(s', a')] - Q(s, a))$$

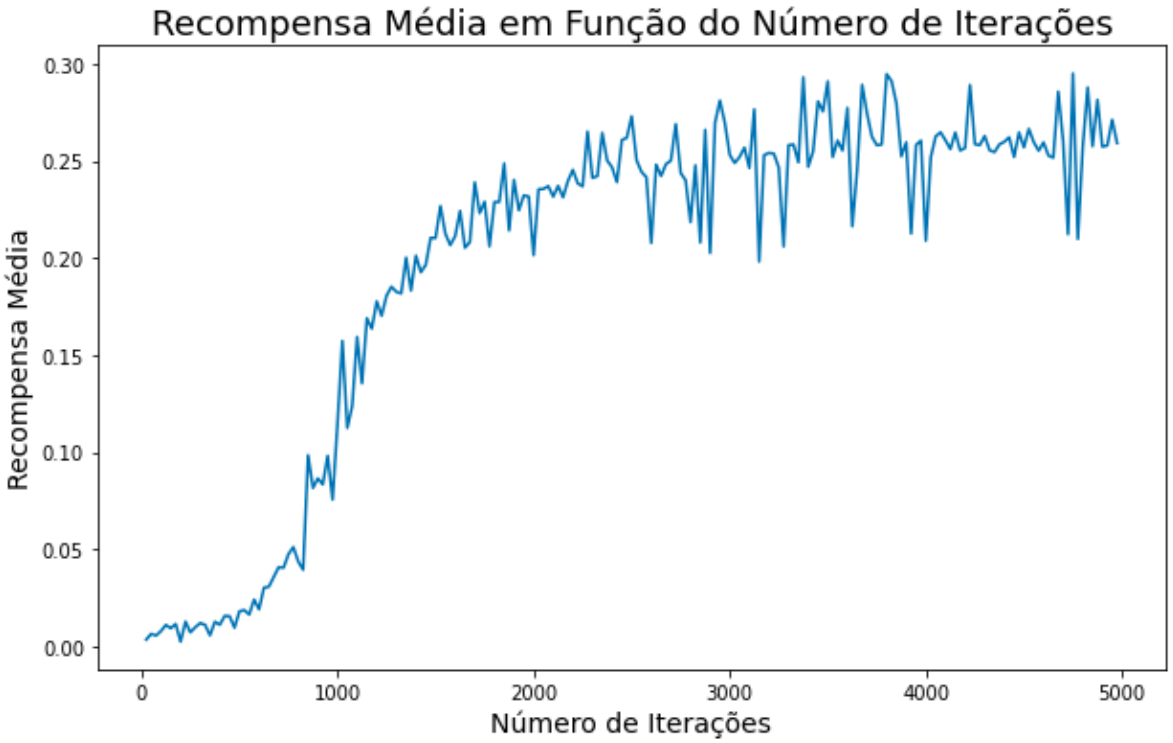
- Por fim, o agente atualiza seu estado atual para o próximo estado observado. Se o estado atual for um estado terminal, o agente retorna a posição inicial.
- Os passos são repetidos até que o agente atinja o número máximo de iterações definido inicialmente.

4) Resultados

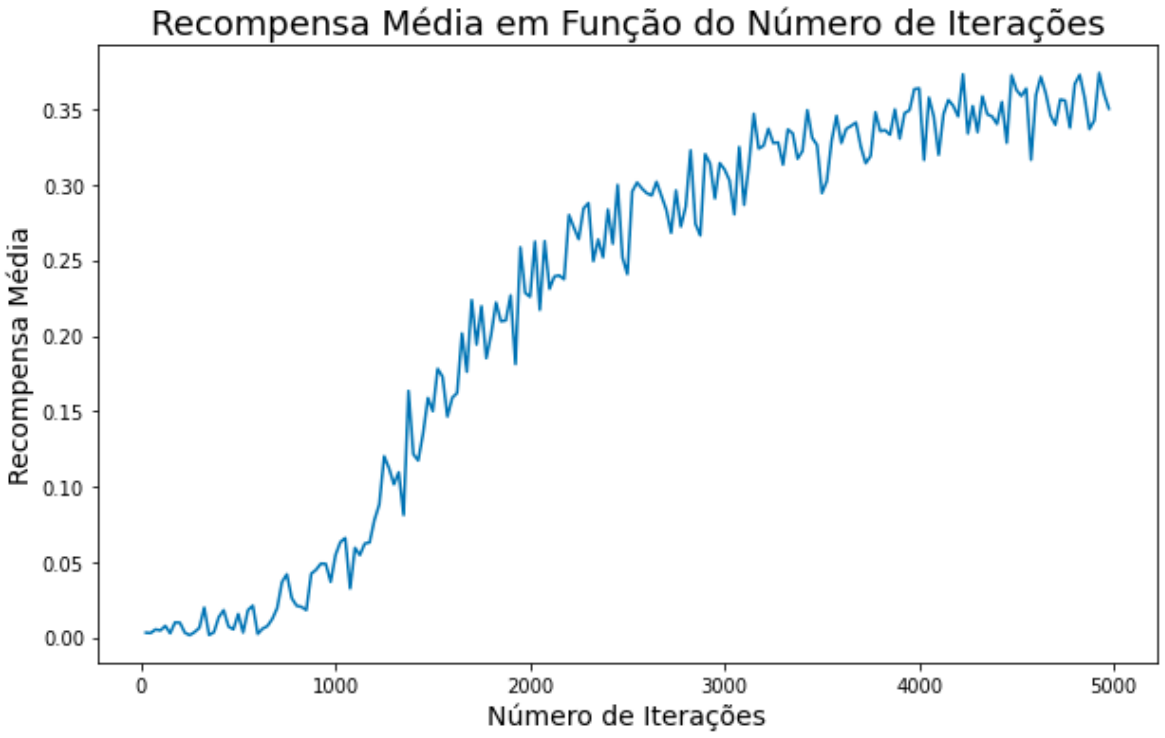
A seguir, há as tabelas de recompensa média para diversos parâmetros de execução em função do número de iterações para o seguinte ambiente:



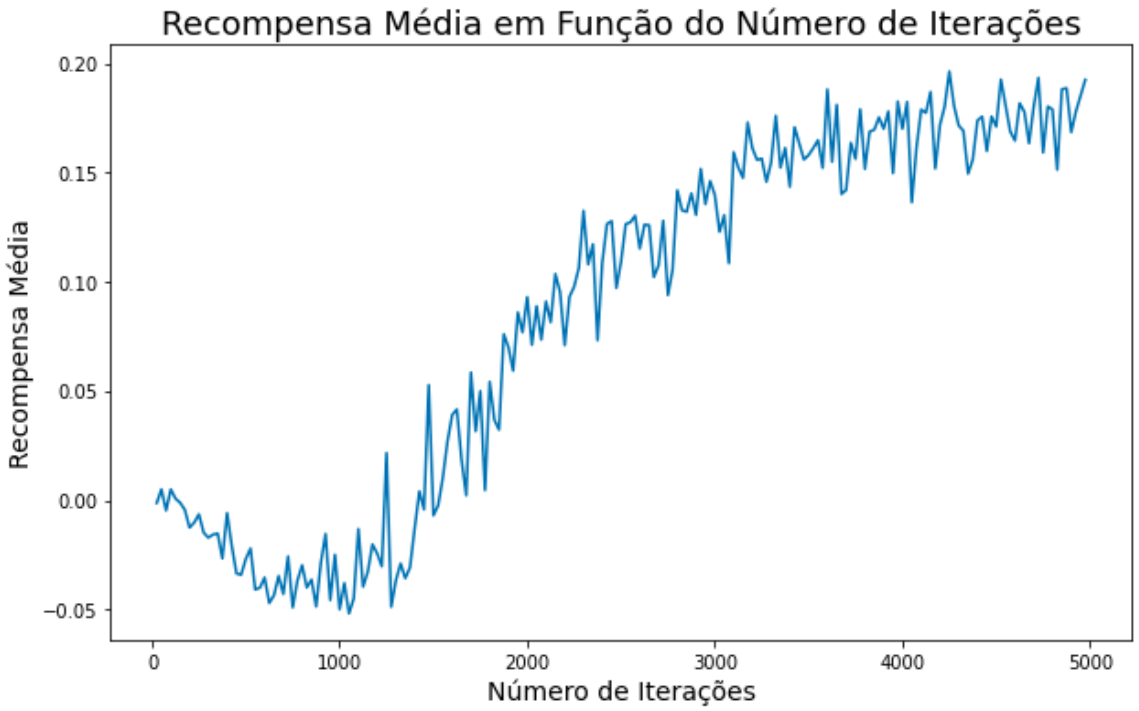
Taxa de aprendizado = 0.1 , Desconto = 0.9 , Recompensa = -0.06



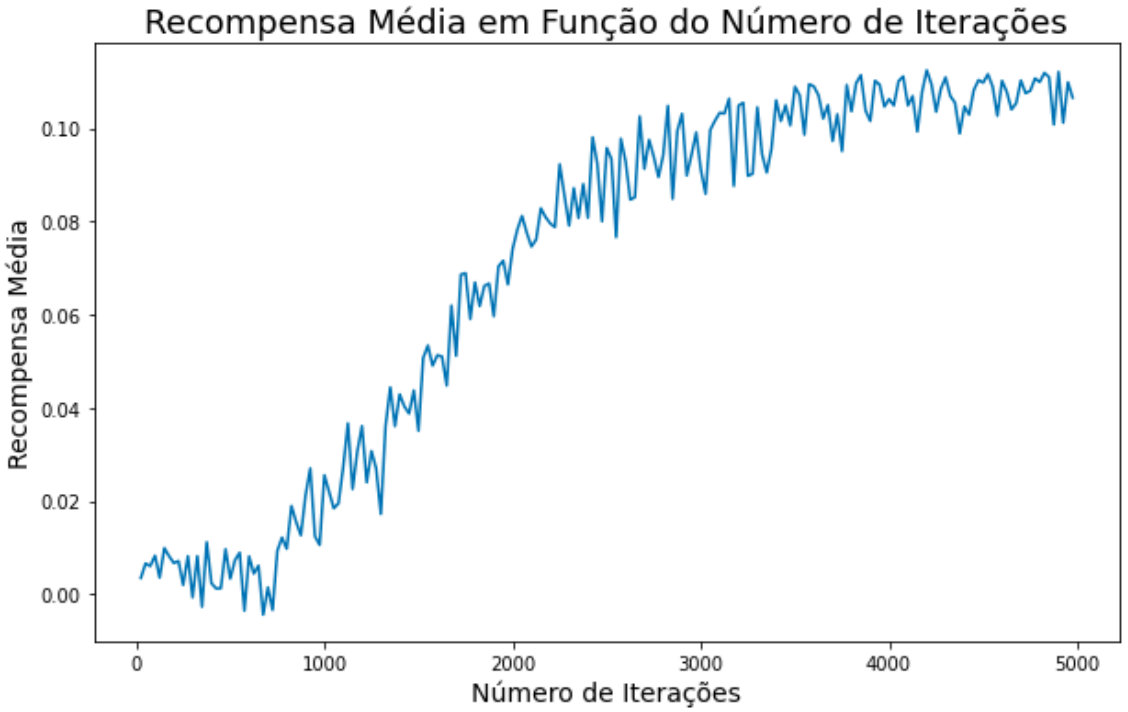
Taxa de aprendizado = 0.1 , Desconto = 0.9 , Recompensa = -0.06, Epsilon = 0.5



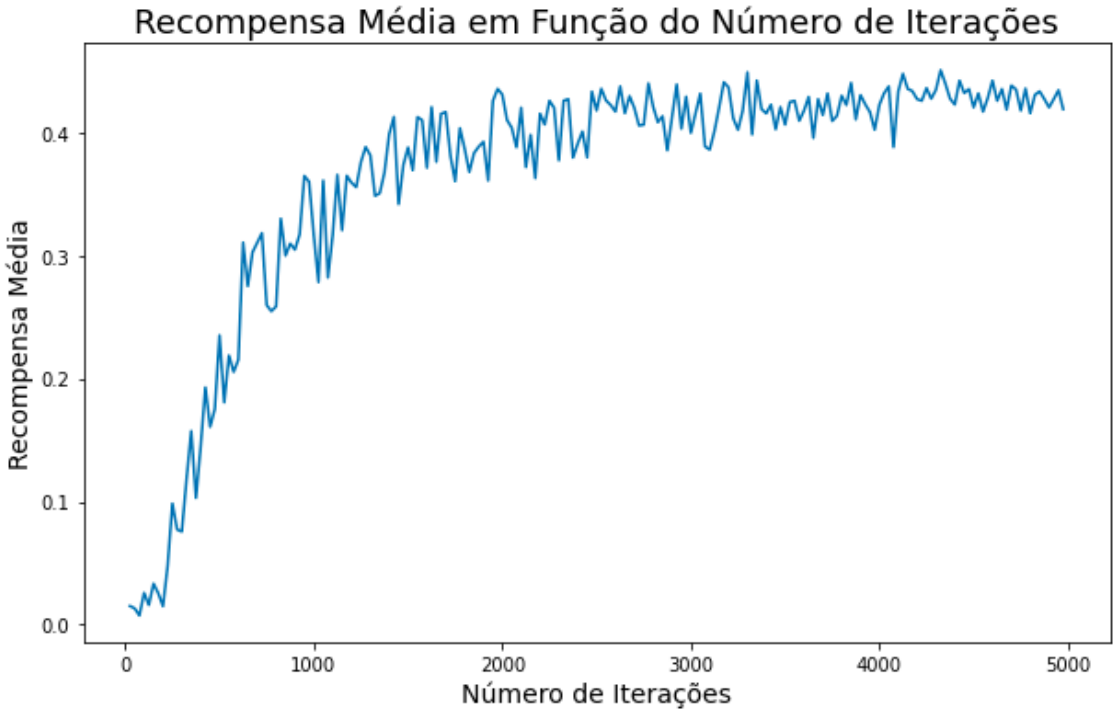
Taxa de aprendizado = 0.1 , Desconto = 0.9 , Recompensa = -0.15, Epsilon = 0.5



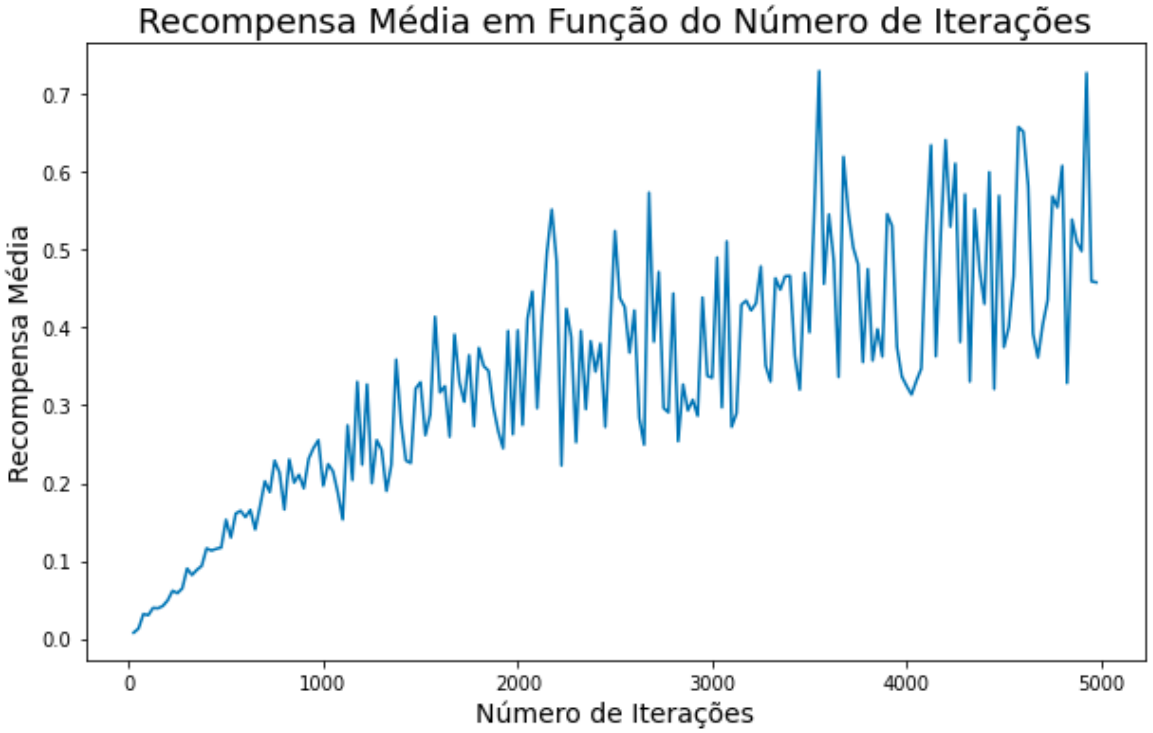
Taxa de aprendizado = 0.1 , Desconto = 0.5 , Recompensa = -0.06, Epsilon = 0.5



Taxa de aprendizado = 0.40, Desconto = 0.9, Recompensa = -0.06, Epsilon = 0.5

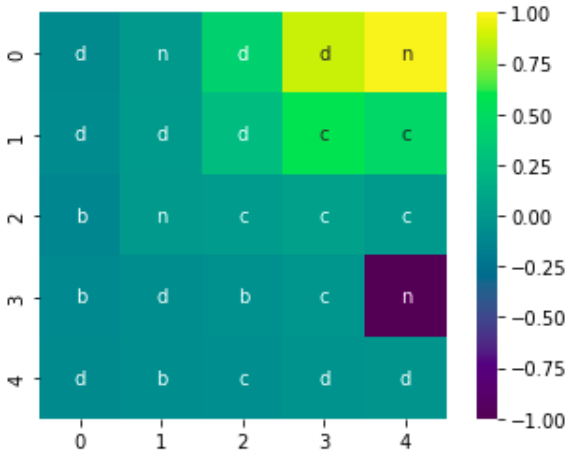


Taxa de aprendizado = 0.1, Desconto = 0.9, Recompensa = 0.10, Epsilon = 0.5

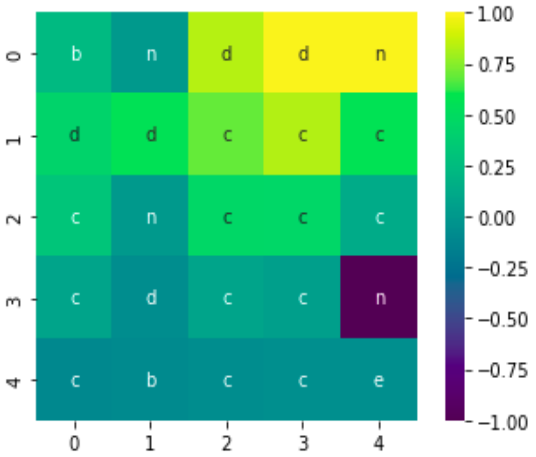


Agora, para o mesmo ambiente, tem se as imagens da melhor ação em cada estado para número de iterações igual a 1000, 2500, 5000 e 10000. O seguintes parâmetros foram usados:
Taxa de aprendizado = 0.1 , Desconto = 0.9 , Recompensa = -0.06, Epsilon =0.5

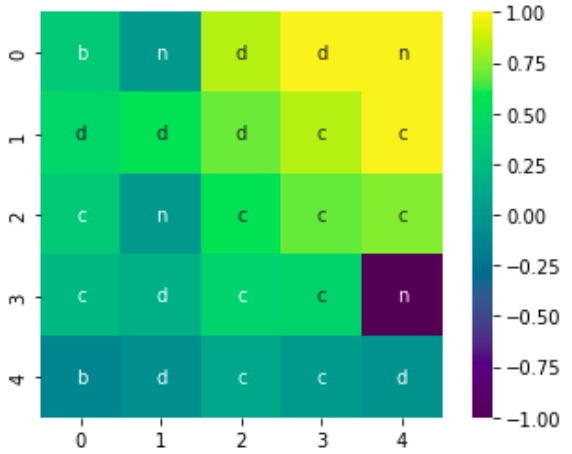
iterações = 1000



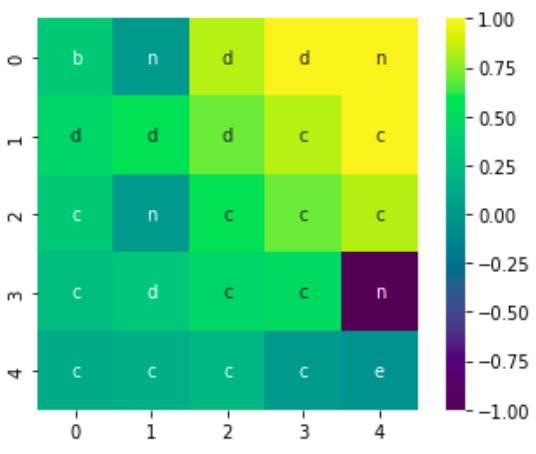
iterações = 2500



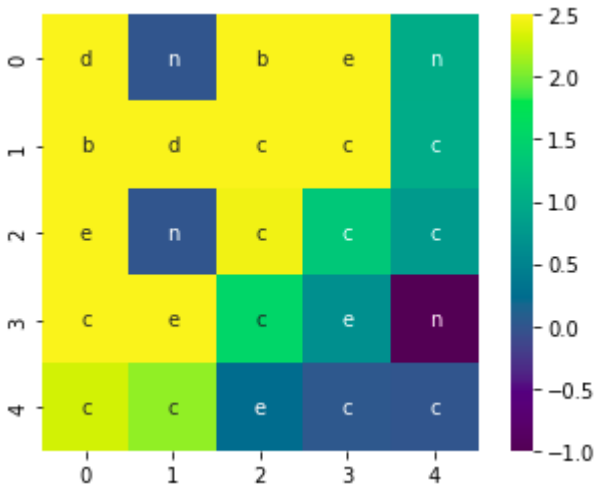
iterações = 5000



iterações = 10000



Por fim, uma imagem dos mesmos parâmetros anteriores, exceto pela recompensa que é positiva (0.25) e para 10000 iterações:



5) Discussão

A seguir, há uma discussão sobre cada um dos gráficos gerados:

Taxa de aprendizado = 0.1, Desconto = 0.9, Recompensa = -0.06

Com essa configuração, espera-se que o agente aprenda lentamente, pois a taxa de aprendizado é relativamente baixa. O valor de desconto alto determina que o agente valorize as recompensas futuras em comparação com as recompensas imediatas, enquanto a recompensa negativa indica que o agente deve procurar a saída do ambiente o mais rápido possível.

Taxa de aprendizado = 0.1, Desconto = 0.9, Recompensa = -0.06, Epsilon = 0.5

A inclusão do valor de epsilon igual a 0.5 significa que o agente seguirá uma política epsilon-greedy, explorando e explotando em proporções iguais. O gráfico gerado ficou bem parecido com o primeiro, mas é possível notar uma recompensa média maior ao final das iterações.

Taxa de aprendizado = 0.1, Desconto = 0.9, Recompensa = -0.15, Epsilon = 0.5:

A redução da recompensa para -0.15 torna o ambiente mais desafiador, pois o agente recebe uma penalidade maior por cada decisão errada no ambiente. Isso resultou em um aprendizado mais lento e em uma recompensa média mais baixa. Inclusive, no ínicio nota-se uma tendência de recompensa média negativa, que logo é compensada quando o agente se adapta ao ambiente.

Taxa de aprendizado = 0.1, Desconto = 0.5, Recompensa = -0.06, Epsilon = 0.5:

A redução do fator de desconto para 0.5 significa que o agente atribui menos importância às recompensas futuras em relação às imediatas. Isso pode levar a um comportamento mais impulsivo do agente, buscando recompensas imediatas em vez de maximizar as recompensas totais a longo prazo. A recompensa média foi afetada, tendo valor menor que os anteriores, pois o agente pode não ser capaz de encontrar uma política ideal devido à baixa consideração pelas recompensas futuras.

Taxa de aprendizado = 0.4, Desconto = 0.9, Recompensa = -0.06, Epsilon = 0.5:

Com uma taxa de aprendizado mais alta, o agente aprenderá mais rapidamente e poderá convergir para uma política ótima em menos iterações, que foi o que de fato ocorreu. A recompensa média foi maior em comparação com

configurações de taxa de aprendizado mais baixas, devido ao agente se adaptar mais rapidamente ao ambiente.

Taxa de aprendizado = 0.1, Desconto = 0.9, Recompensa = 0.10, Epsilon = 0.5:

A recompensa positiva de 0.10 indica que o agente recebe recompensa por ficar rodeando pelo ambiente indefinidamente. Isso incentivará o agente a evitar estados terminais para maximizar sua recompensa total. A recompensa média nesse caso pode ser relativamente alta e bastante variável, como pode ser observado no gráfico gerado para essa configuração.

Já nas imagens geradas, pode-se notar a evolução das melhores ações para cada estado com o aumento das iterações.

Para 1000 iterações, observa-se que o agente está começando a explorar o ambiente e ainda não possui um conhecimento completo das melhores ações em cada estado. À medida que o número de iterações aumenta para 2500, o agente começa a tomar decisões mais precisas e a identificar as melhores rotas para alcançar o estado terminal correto.

Por fim, para os números de iterações igual a 5000 e 10000, nota-se que as imagens ficaram bem semelhantes e guiam o agente para a o estado terminal correto. A principal diferença foi nos cantos inferiores, em que para 5000 ainda nota-se decisões que fazem com que o agente fique no mesmo estado, enquanto para 10000 as ações ideais possibilitam que o agente saia das laterais do ambiente.

A imagem da configuração com recompensa positiva mostra uma situação diferente, em que o agente, como já expresso anteriormente, evita estados terminais a fim de aumentar seus ganhos.

6) Conclusão

O algoritmo Q-Learning demonstrou ser uma abordagem eficaz para resolver problemas de aprendizado por reforço em ambientes discretos.

Além disso, a realização desse trabalho foi uma boa oportunidade de colocar em prática e compreender os fundamentos do aprendizado por reforço e do algoritmo Q-Learning. Por meio da implementação e dos experimentos realizados, foi possível observar os resultados do treinamento do agente e compreender os princípios e desafios envolvidos nessa abordagem.