Trabalho Prático 2

Sarah Oliveira Elias - 2018048478

12/12/2022

1. Introdução

O trabalho prático tem como tema o aprendizado por reforço. O problema proposto foi implementar o método Q-Learning. Dado um mapa com características específicas de terreno e um ponto inicial, o objetivo era treinar o agente para gerar uma política que deveria ser retornada como solução do problema. Para encontrar a solução, foram desenvolvidos três métodos diferentes de Q-Learning: Q-Learning padrão sem modificações, Q-Learning com recompensas positivas, Q-Learning com ambiente estocástico.

Com esse trabalho prático foram praticados conceitos de programação abordados durante o curso, isto é, modelagem de problemas de aprendizado por reforço e algoritmos e estruturas de dados necessários para resolvê-los.

2. Modelagem

2.1. Conjunto de Estados

Os estados do problema são todas as posições da matriz bidimensional que representa o mapa, exceto aquelas que apresentam o símbolo @, pois representam uma posição de terreno intransponível. A recompensa associada a cada estado é dada pelo terreno do estado. O estado inicial é a posição no mapa indicada pelas coordenadas recebidas como argumento pelo programa.

2.2. Conjunto de ações aplicáveis naquele estado

O agente pode se mover pelo mapa realizando os movimentos cima, baixo, direita e esquerda. Ações que resultam em posições fora dos limites do mapa ou em estados que tenham o símbolo @ são válidas, porém, após executá-las, o agente permanece no mesmo estado.

2.3. Agente

Dado um estado, o agente se move para um de seus vizinhos através dos movimentos possíveis. O método de exploração utilizado foi o ϵ-greedy com valor $\epsilon = 0.1$. Isso significa que em 90% das vezes, a escolha do próximo estado se dá pela avaliação da função de valor das ações de cada estado. Nesse caso, o agente escolhe se mover para o estado que possui a ação com maior valor entre todos os estados alcançáveis. Em 10% das vezes o agente faz uma escolha aleatória do próximo estado a ser explorado.

Cálculo da função de valor de cada estado se dá através da soma da recompensa real que o agente recebe neste estado com a estimativa das recompensas futuras que podem ser recebidas a partir dele.

2.4. Ambiente

Nesse trabalho, exploramos um ambiente determinístico e um estocástico. No determinístico, a ação escolhida pelo agente sempre é executada sem nenhuma interferência. Já no estocástico, a ação escolhida é executada em 80% das vezes. A cada escolha, há 20% de chance de uma ação perpendicular à escolhida ser executada pelo ambiente. Em 10% das vezes a ação é desviada para a esquerda e em 10% ela é desviada para a direita.

3. Implementação

O programa foi desenvolvido na linguagem C++, compilada pelo compilador G++ da GNU Compiler Collection.

3.1. Mapa e recompensas

O mapa foi representado por uma matriz de dimensões H x W que armazena as recompensas de cada terreno. Cada posição da matriz guarda um valor de ponto flutuante que corresponde à recompensa do terreno.

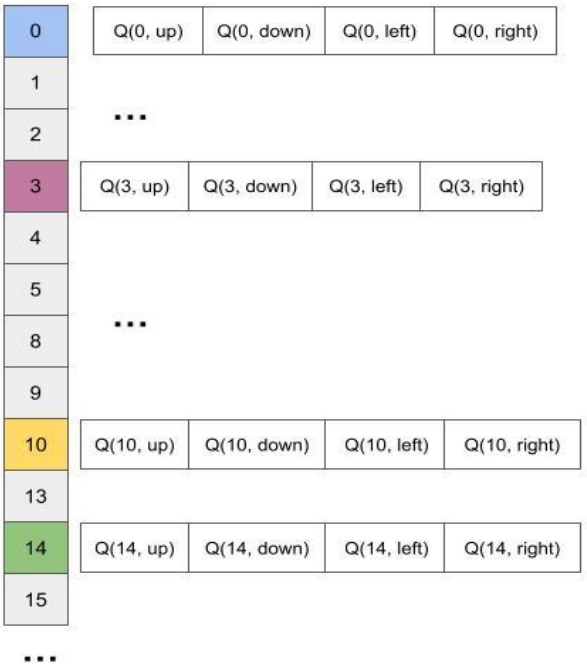
Essa matriz foi implementada usando vector da Biblioteca Padrão do C++ (C++ Standard Library).

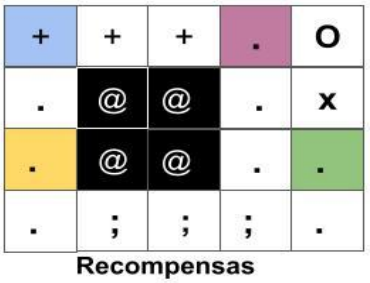
3.2. Estados

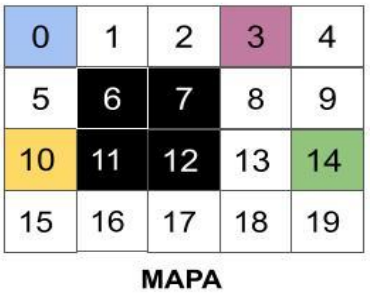
Os estados foram representados por um vetor de tamanho 4. Para um estado s, os índices 0, 1, 2, 3 do seu vetor guardam os valores relativos à função de valor das ações possíveis em s, isto é, os valores Q(s, up), Q(s, down), Q(s, left) e Q(s, right), respectivamente.

O conjunto de todos esses vetores, cada um representando um estado, são armazenados sequencialmente em um vetor de tamanho H*W. Esses vetores também foram implementados usando vector da biblioteca padrão.

Como cada estado está associado a uma posição do mapa, ele foi indexado de acordo com as posições (x, y) da matriz que representa o mapa. A maneira com que os índices foram atribuídos é mostrada na imagem abaixo:







4. Descrição dos algoritmos.

4.1 Q-Learning padrão

Q-learning é um algoritmo de aprendizagem por reforço no qual o agente inicia no ambiente sem nenhum conhecimento prévio e tem como objetivo gerar uma política para o mapa desse ambiente através da exploração de seus estados.

O algoritmo implementado para esse trabalho recebe um mapa, no qual cada posição (x,y) representa um estado com uma recompensa associada. O valor dessa recompensa é dado pelo tipo de terreno de cada posição (estado) do mapa. O agente explora esses estados e as ações possíveis a partir deles, atualizando a Função de Valor de Ações Q(s,a) a cada passo.

O valor de Q para um estado **s** e uma ação **a** é definido considerando o valor da recompensa real do estado e o valor da estimativa de recompensas futuras a serem recebidas nos estados seguintes. Esse valor é calculado pela função:

$Q(s,a) = Q(s,a) - \alpha [r + \gamma (\max Q(s', a') - Q(s,a)]$

Os valores utilizados na implementação foram:

Taxa de aprendizado $\alpha = 0.1$
Taxa de desconto $\gamma = 0.9$
Método de exploração: ϵ-greedy ($\epsilon = 0.1$)

4.2 Q-Learning com Recompensas positivas

Esse método se diferencia do padrão apenas no valor das recompensas do terreno, que nesse caso são valores positivos, com exceção do terreno de fogo, que é igual a zero. A hierarquia continua igual, isto é, o ‘objetivo’ ainda tem a maior recompensa, o ‘fogo’ tem a recompensa mais baixa e etc.

4.3 Q-Learning em Ambiente Estocástico

Esse método contém uma modificação feita para que exista uma chance de 20% de que a ação escolhida pelo agente não seja executada. Em 10% das vezes, uma ação perpendicular à esquerda da ação escolhida é executada, e em 10% das vezes uma ação perpendicular à direita é executada. Em 80% das vezes, a ação escolhida é executada.

5. Análise dos Resultados

Os métodos padrão e estocásticos geram políticas eficientes, sendo que o padrão gera políticas ótimas na maioria dos casos. Já o Q-Learning com recompensas positivas não foi capaz de gerar políticas satisfatórias.

Padrão
O Q-Learning padrão apresentou o melhor desempenho de todos. Esse método acha uma política razoável sempre que o número de passos é suficiente.Ele nem sempre explora todos os estados de forma satisfatória, porém, dado um início, sempre encontra um caminho bom para o objetivo a partir desse início.

A partir de 1.000.000 passos, o algoritmo parece retornar a política ótima na maioria das vezes. A política melhora quando o ponto inicial se encontra mais distante do objetivo, pois isso permite que o agente explore o mapa de forma mais eficiente.

Estocástico
As políticas apresentadas por esse método foram iguais ou muitas vezes um pouco menos eficientes do que aquelas apresentadas pelo padrão. Na maioria das execuções, a política leva ao objetivo a partir de todos os estados do mapa sem passar por fogo, mas os caminhos indicados por ela nem sempre são ótimos.

Isso ocorre porque a mudança feita aumenta a incerteza do agente de ir para o estado escolhido a partir da ação escolhida, e a consequência disso é que os valores de Q aumentam mais lentamente. Os estados menos explorados são os que mais evidenciam essa característica. Assim, a política gerada por esse método é menos eficiente pois a exploração do agente do ambiente é mais imprecisa do que a do método padrão.

Recompensas positivas
O Q-Learning com Recompensas positivas foi o método com o pior desempenho. Ao adicionar a mudança para recompensas positivas, as políticas geradas se tornam erradas; diferente das políticas retornadas pelo método padrão, elas não mostram um caminho até o

objetivo (algumas políticas formam um desenho que causam a impressão de que o agente está tentando fugir do objetivo).

Isso ocorre porque o agente recebe recompensa positiva para permanecer nos estados que não são terminais. A política é orientada para que o agente não saia do ambiente porque esse é o comportamento que recebe maior recompensa.

ϵ-greedy
Mudando o valor de ϵ do método ϵ-greedy de 0.1 para 0.2 já temos uma mudança considerável. Com ϵ = 0.2 e número de passos igual a 100.000, tanto o Q-Learning padrão quanto o estocástico passam a retornar sempre a melhor política. A mudança no método de recompensas positivas não é muito significativa.

Isso ocorre porque a mudança permite que o agente explore mais o ambiente, mudando de estado de forma aleatória com mais frequência e, consequentemente, conseguindo avaliar todos os estados com mais precisão.

6. Conclusão

Esse trabalho foi uma ótima oportunidade de praticar os conceitos aprendidos até agora durante o curso. Os resultados ficaram dentro do esperado esperado para todos os algoritmos, e as respostas entregues por eles foram satisfatórias no caso do Q-Learning padrão e do Q-Learning estocástico.

7. Referências

[1] Richard S. Sutton and Andrew G. Barto. (2020) Reinforcement Learning - An Introduction. 2th Edition.

[2] Stuart, R. and Peter, N. (2020). Artificial Intelligence – A Modern Approach. 4th Edition, Prentice Hall

8. Anexos

Listagem dos programas:
main.cpp

Compilação:
g++ main.cpp -o qlearning

Exemplo de como rodar:
./qlearning maze.map stochastic 1 3 300000