

Universidade Federal do Rio de Janeiro  
Instituto Alberto Luiz Coimbra de  
Pós-Graduação e Pesquisa de Engenharia



Programa de Engenharia de Sistemas e  
Computação

CPS863 - Aprendizado de Máquina  
Prof. Dr. Edmundo de Souza e Silva  
(PESC/COPPE/UFRJ)

*Lista de Exercícios 6*

Luiz Henrique Souza Caldas  
email: lhscaldas@cos.ufrj.br

14 de dezembro de 2024

## Questão 1

*Value Iteration*, *Policy Iteration* e *Q-Learning* são algoritmos utilizados para encontrar a política ótima em problemas de decisão sequencial, como um Processo de Decisão de Markov (MDP). A diferença entre eles é a forma como a política ótima é encontrada. A descrição de cada um deles abaixo e as equações seguem a notação do livro *Reinforcement Learning: An Introduction* de Sutton e Barto [1].

### Value Iteration

Calcula iterativamente a função de valor  $V(s)$  para cada estado  $s$  até convergir para a função de valor ótima  $V^*(s)$ . A função de valor é calculada tornando a equação de Bellman de otimalidade em uma regra de atualização iterativa:

$$V_{k+1}(s) = \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V_k(s')] \quad (1)$$

onde  $V_k(s)$  é a função de valor no passo  $k$ ,  $p(s', r|s, a)$  é a probabilidade de transição para o estado  $s'$  e recompensa  $r$  dado o estado  $s$  e ação  $a$  e  $\gamma$  é o fator de desconto, que regula a importância dada as recompensas futuras.

A convergência da função de valor é dada pela condição de parada:

$$\Delta = \max_s |V_{k+1}(s) - V_k(s)| < \theta \quad (2)$$

onde  $\theta$  é um pequeno limiar (*threshold*), que determina a acurácia da convergência. Ao ser atingida esta condição, podemos considerar que a função de valor ótima  $V^*(s)$  foi encontrada:

$$V^*(s) \approx V_k(s) \quad (3)$$

Após a convergência da função de valor, a política ótima  $\pi^*(s)$  é obtida a partir da função de valor ótima:

$$\pi^*(s) = \arg \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V^*(s')] \quad (4)$$

onde  $\arg \max$  é o operador que retorna o argumento que maximiza a função.

### Policy Iteration

Calcula iterativamente a política ótima  $\pi^*(s)$  em duas etapas: **avaliação** e **melhoria da política**.

- Na etapa de avaliação, a função de valor  $V(s)$  é calculada para a política atual  $\pi(s)$  a partir da equação de Bellman:

$$V_{k+1}(s) = \sum_{s',r} p(s', r|s, \pi(s))[r + \gamma V_k(s')] \quad (5)$$

onde  $V(s)$  é a função de valor para o estado  $s$ ,  $p(s', r|s, \pi(s))$  é a probabilidade de transição para o estado  $s'$  e recompensa  $r$  dado o estado  $s$  e ação  $\pi(s)$  e  $\gamma$  é o fator de desconto. Esse processo é repetido até ser atingido o critério de convergência  $\Delta = \max_s |V_{k+1}(s) - V_k(s)| < \theta$ .

- Na etapa de melhoria da política, feita após a avaliação da função de valor, a política é atualizada para a ação que maximiza a função de valor:

$$\pi_{k+1}(s) = \arg \max_a \sum_{s', r} p(s', r|s, a) [r + \gamma V_k(s')] \quad (6)$$

O algoritmo inicializa com uma política  $\pi(s)$  arbitrária e continua iterando entre a avaliação e melhoria da política até que a política não mude mais. Neste ponto, a política ótima  $\pi^*(s)$  foi encontrada.

## Q-Learning

Calcula a política ótima  $\pi^*(s)$  aprendendo a função de ação-valor  $Q(S, A)$ , que estima o retorno esperado ao tomar a ação  $a$  no estado  $s$ , sem depender de conhecimento prévio de um modelo do ambiente (transições e recompensas). O algoritmo atualiza a função de ação-valor iterativamente a partir da equação de Bellman para a função de ação-valor:

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha [R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \quad (7)$$

onde a ação  $A$  é escolhida de acordo com uma política de exploração,  $\alpha$  é a taxa de aprendizado,  $R$  é a recompensa imediata,  $\gamma$  é o fator de desconto e  $S_{t+1}$  é o estado resultante da ação  $A_t$  no estado  $S_t$ .

A política de exploração é diferente da política ótima, e é usada para explorar o ambiente e evitar a convergência prematura para uma política subótima. Isso faz com que o algoritmo *Q-Learning* seja considerado um algoritmo de aprendizado por reforço *off-policy*. Uma política de exploração comum é a política  $\epsilon$ -gulosa, que escolhe a ação que maximiza a função de ação-valor com probabilidade  $1 - \epsilon$  e uma ação aleatória com probabilidade  $\epsilon$ .

O termo  $R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)$  é o erro TD (*Temporal Difference*), que é usado para atualizar a função de ação-valor. Isso faz com que o algoritmo *Q-Learning* seja incluído na categoria de métodos de aprendizado por reforço baseados em diferenças temporais (*Temporal Difference Learning*).

$Q(S, A)$  é inicializado arbitrariamente e atualizado a cada passo da simulação, até que a função de ação-valor convirja para a função de ação-valor ótima  $Q^*(S, A)$ . A política ótima  $\pi^*(s)$  é obtida a partir da função de ação-valor ótima:

$$\pi^*(s) = \arg \max_a Q^*(s, a) \quad (8)$$

## Comparação

A principal diferença entre *Value Iteration* e *Policy Iteration* é que o primeiro calcula a função de valor diretamente, iterando sobre ela até encontrar o valor ótimo, para então derivar a política ótima. Já o segundo calcula a função de valor para a política atual e, em seguida, atualiza a política

para a ação que maximiza a função de valor. Este processo é repetido até que a política não mude mais. Já o *Q-Learning* se diferencia dos outros dois por não precisar de um modelo do ambiente para calcular a política ótima. Ele aprende a política ótima interagindo com o ambiente (ou uma simulação deste), atualizando a função de ação-valor. A política ótima é obtida escolhendo a ação que maximiza o valor estimado.

## Códigos

Os códigos utilizados para a resolução dos exercícios estão disponíveis no repositório do GitHub:  
<https://github.com/lhscaldas/cps863/>

## Referências

- [1] SUTTON, R. S.; BARTO, A. G. *Reinforcement Learning: An Introduction*. 2nd. ed. Cambridge, MA: MIT Press, 2018. ISBN 978-0262039246.