MAC0434 e MAC5788 - Planejamento e Aprendizado por Reforço (2023)

EP 2

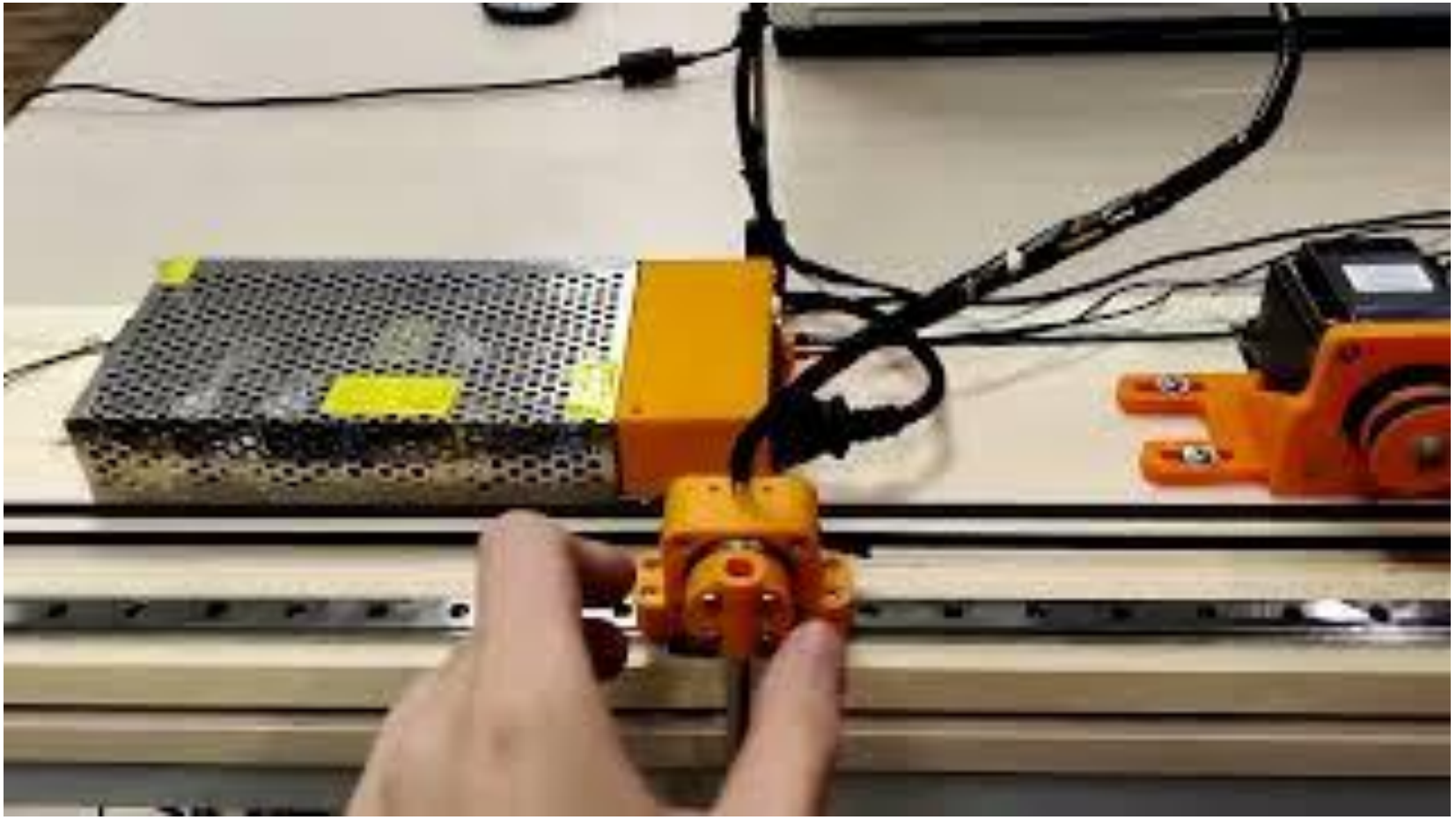
SARSA e Q-learning

Tarefa

- Implementar os algoritmos SARSA e Q-learning
- Grupos de 2 alunos
- Domínio: **Cart Pole** do OpenAI-gym
- Entrega de um relatório com uma análise comparativa de desempenho dos 2 algoritmos

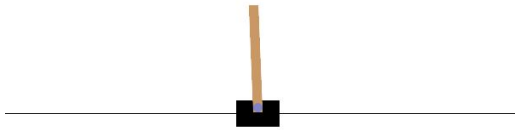
Cart Pole - Exemplo de um protótipo

Veja o link: https://www.youtube.com/watch?v=rvKnKtFgE7U



Cart Pole

- Cart Pole é um ambiente que envolve um pêndulo invertido conectado a um carrinho capaz de se mover horizontalmente para a esquerda ou direita, com o objetivo de manter o pêndulo equilibrado a 90 graus.
- Neste EP, você deve utilizar este ambiente disponível através da biblioteca python `gymnasium`. **Sua documentação pode ser encontrada em:** https://gymnasium.farama.org/environments/classic_control/cart_pole/
- Utilize a versão **`CartPole-v1`**
- O EP deve ser implementado na linguagem python



Cart Pole - Gymnasium

Observações

Num	Observation	Min	Max
0	Cart Position	-4.8	4.8
1	Cart Velocity	-Inf	Inf
2	Pole Angle	~ -0.418 rad (-24°)	~ 0.418 rad (24°)
3	Pole Angular Velocity	-Inf	Inf

Cart Pole - Gymnasium

Ações

- Mover para a direita
- Mover para a esquerda

Fim de um episódio

- O ângulo do pêndulo é maior que ±12°
- A posição do carrinho é maior que ±2,4
- A duração do episódio é maior que 500

Recompensas

- A cada ação realizada no ambiente, o sistema devolve uma recompensa, que pode ser 1 se a ação realizada pontuou de forma a alcançar o objetivo do ambiente, ou 0 se a ação realizada falhou.

Cart Pole - Exemplo de uso da biblioteca Gymnasium

Tupla devolvido a cada interação (execução de ação) com o simulador:

(observation, reward, terminated, truncated, info)

observation = próximo estado (lista [0..3])

reward = recompensa recebida

terminated = indicação de que o episódio foi encerrado (*Pole Angle is greater than ±12° or Cart Position is greater than ±2.4*)

truncated = indicação se o episódio finalizou (*Episode length is greater than 500*)

Documentação: https://gymnasium.farama.org/content/basic_usage/

Cart Pole - Exemplo de uso da biblioteca Gymnasium

```
import gymnasium as gym

env = gym.make("CartPole-v1")
observation, info = env.reset() # ver especificação da inicialização do ambiente


# exemplo de interação com o ambiente usando uma política aleatória
for iteration in range(100):
   action = env.action_space.sample()  # seleciona uma ação aleatória
   observation, reward, terminated, truncated, info  = env.step(action) #executa a
ação

   print('Current iteration: ', iteration)
   print('Current state: ', observation)
   print('\tCart Position\t\t', observation[0])
   print('\tCart Velocity\t\t', observation[1])
   print('\tPole Angle\t\t', observation[2])
   print('\tPole Angular Velocity\t', observation[3])
   print('Action taken: ', action)

   if terminated or truncated:
       observation, info  = env.reset()
       break # finaliza as iterações se o episódio chegou ao fim
env.close()
```

Algoritmo SARSA: on-policy control (S, A, R, S', A')

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

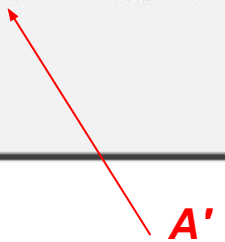
Loop for each episode:
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') - Q(S, A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

Q-learning para controle off-policy

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until S is terminal

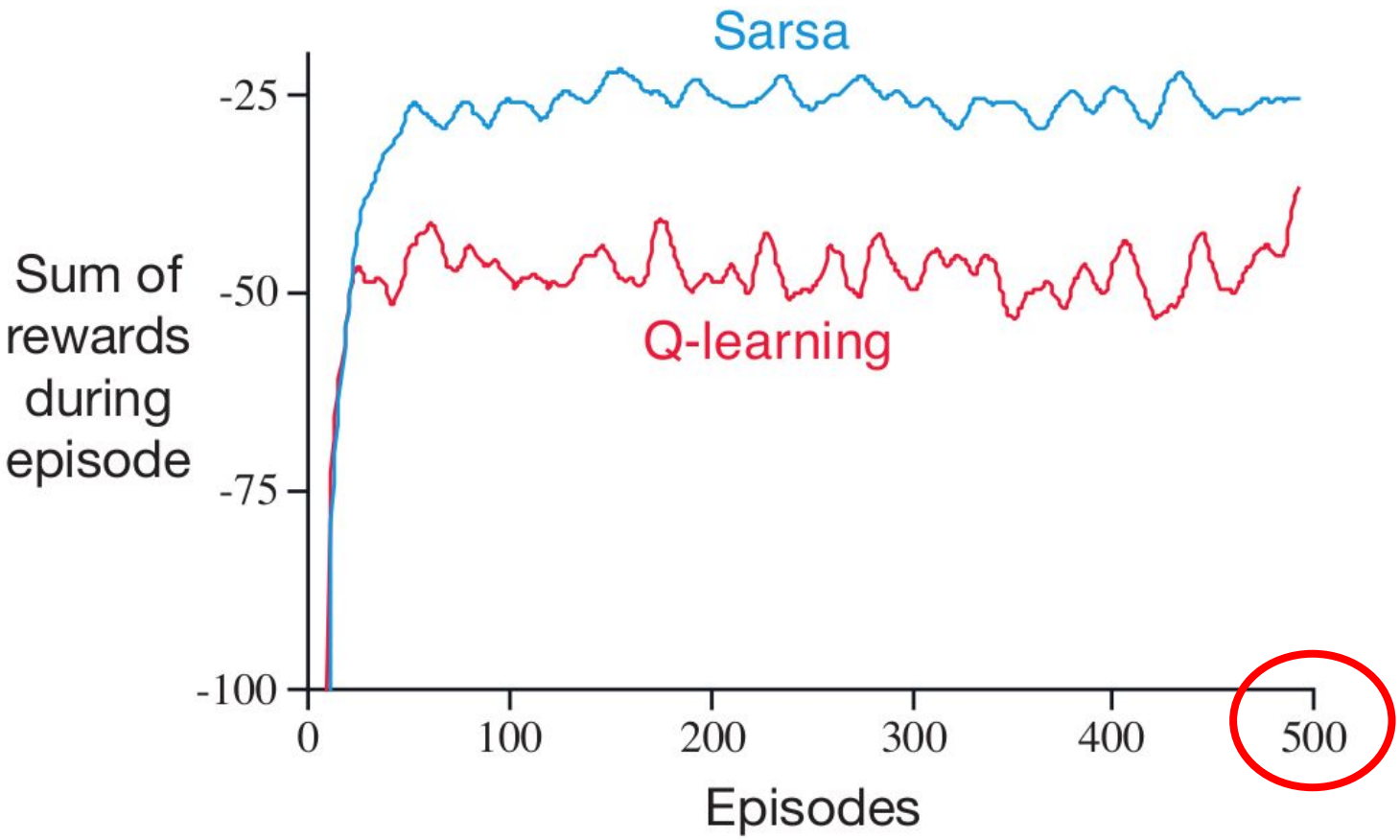


Implementação

- inicialização de valores aleatórios
- estrutura de dados para armazenar as estimativas da função Q(s,a)
- geração da próxima **ϵ-greedy** action
- valores dos parâmetros envolvidos nos algoritmos:
  - Fator de desconto: γ (entre 0 e 1, recomenda-se usar valores altos, por exemplo 0.8 ou 0.9)
  - Taxa de aprendizado: **α**
  - Taxa de exploração: **ϵ**
- condições de fim de episódio:
  - ângulo do pêndulo maior que ±12°
  - posição do carrinho maior que ±2,4
  - duração do episódio maior que 500
- plote das curvas dos resultados
  - por exemplo, usando a soma das recompensas acumuladas em cada episódio, etc.
- no. de episódios: o suficiente para observar convergência

Exemplo de um gráfico comparativo

Gráfico extraído do livro: *Reinforcement Learning*. Richard S. Sutton e Andrew G. Barto



Discretização dos estados

No ambiente Cart Pole, os estados são representados por valores *contínuos*. Armazenar esses estados sem a aplicação de técnicas de discretização pode exigir recursos computacionais significativos. Portanto, utilize **técnicas de discretização para a representação dos estados**.

Experimentos

Algumas sugestões do que pode ser explorado nos experimentos:

- variar a dinâmica exploração *x* exploitação
- comparar diferentes técnicas de discretização
- etc

Qual a melhor combinação de parâmetros que você consegue obter com seus algoritmos?

Recomendações

- Código organizado e de fácil compreensão;
- Colocar os nomes dos integrantes dupla em todos os arquivos que serão entregues (incluindo arquivos de código);
- Apresentar de forma clara e detalhada como os experimentos foram realizados (parâmetros, quantidade de execuções, etc).
  - Utilize gráficos e tabelas para ilustrar os resultados e análise dos experimentos.

O que deve ser entregue no e-disciplinas

- Entrega do relatório em formato PDF.
- O relatório deve conter as seguintes seções obrigatórias:
  - seção de introdução (definições e objetivo do trabalho),
  - seção de fundamentos com a especificação dos algoritmos usando uma notação coerente com a empregada nos slides (e/ou livro),
  - seção de experimentos, inicializando com a descrição dos detalhes da sua implementação incluindo valores iniciais dos parâmetros, análise comparativa dos resultados dos experimentos mostrados por gráficos e tabelas.
  - seção de conclusão.
- Arquivos python com todo código implementado.
- Coloque todos os arquivos em uma pasta **.zip** e submeta no moodle.

Material extra

- Implementação do Cart Pole na bibliteca Gymnasuim: https://github.com/Farama-Foundation/Gymnasium/blob/main/gymnasium/envs/classic_control/cartpole.py