

Algoritmo de Monte-Carlo

Tentativa e erro na pratica

O que é?

Método de Monte-Carlo

- Família de métodos que aprendem $v^*(s)$ ou $q^*(s, a)$ baseado em amostras obtidas por experiências com o ambiente



O que é?

Método de Monte-Carlo

- O agente usa π para um completar um episódio completo, gerando:

$$S_0, A_0, R_1, S_1, A_1, R_2, \dots, S_{T-1}, A_{T-1}, R_T$$

O que podemos calcular com esse histórico?

O que é?

Método de Monte-Carlo

- A estimativa de valores se dá obtendo amostragens do ambiente, gerando n retornos G que serão ponderados

$$v_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$$

$$G_t = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1}$$

$$V_{\pi}(s) = \frac{1}{N} \sum_{k=1}^N G_{s_k}$$

O que é?

Método de Monte-Carlo

- A estimativa de valores se dá obtendo amostragens do ambiente, gerando n retornos G que serão ponderados

$$q_{\pi}(s, a) = \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a]$$

$$G_t = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1}$$

$$Q_{\pi}(s, a) = \frac{1}{N} \sum_{k=1}^N G_{s, a_k}$$

O que é?

Por que?

- Essa média é calculada pois de acordo com a lei dos números grandes, quanto mais amostras de G tiver, mais próximo do real valor de $v_{\pi}(s)$ se chega

$$P(\lim_{n \rightarrow \infty} \bar{G}_s = v_{\pi}(s)) = 1$$

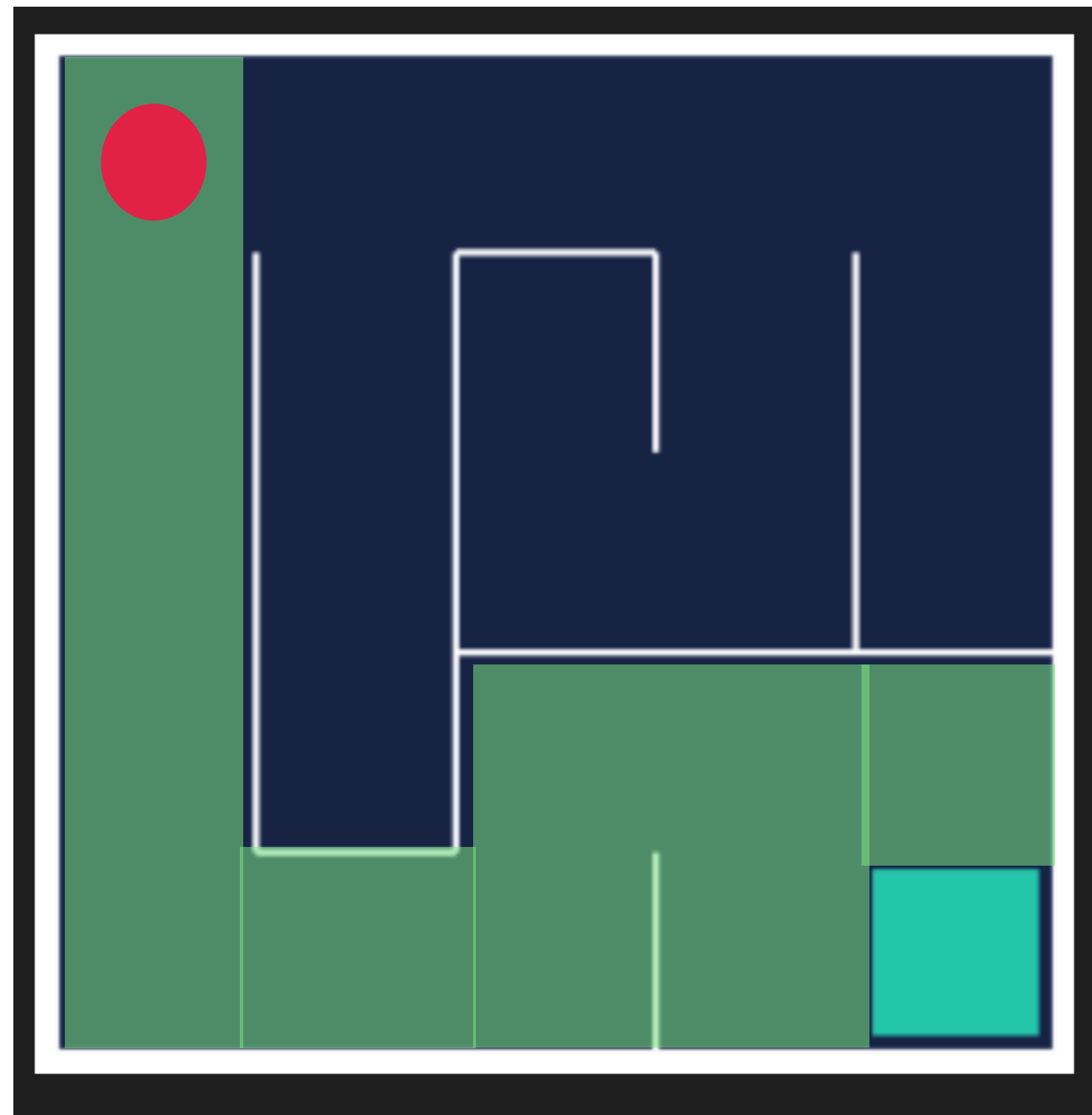
Por que usamos Métodos de Monte-Carlo?

Por que?

- A estimativa do valor de um estado não depende dos outros
- O custo de estimar um valor de um estado é independente do numero total de estados
- Não necessita do modelo completo do ambiente!

Por que usamos Métodos de Monte-Carlo?

- É possível focar no espaço de solução apenas



Como um MMC resolve um problema de RL

- Inicia-se com um episódio usando uma politica arbitraria π gerando uma trajetória inicial:

$$S_0, A_0, R_1, S_1, A_1, R_2, \dots, S_{T-1}, A_{T-1}, R_T$$

- Desta trajetória, calcula-se retornos para cada momento t

$$\begin{aligned} G_t &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-t-1} R_T \\ G_{t+1} &= R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \dots + \gamma^{T-t-1} R_T \\ &\dots \end{aligned}$$

- Estima-se $V(s)$ e atualiza-se π

Como um MMC resolve um problema de RL

- Para atualizar π , se necessita saber os valores de $v(s)$, mas não é possível obtê-los

$$\pi'(s) = \operatorname{argmax}_a \sum_{s',r} p(s', r | s, a) [r + \gamma v_{\pi}(s')]$$

- Porém se há valores $q(s, a)$

$$q_{\pi}(s, a) = \sum_{s',r} p(s', r | s, a) [r + \gamma v_{\pi}(s')]$$

Como um MMC resolve um problema de RL

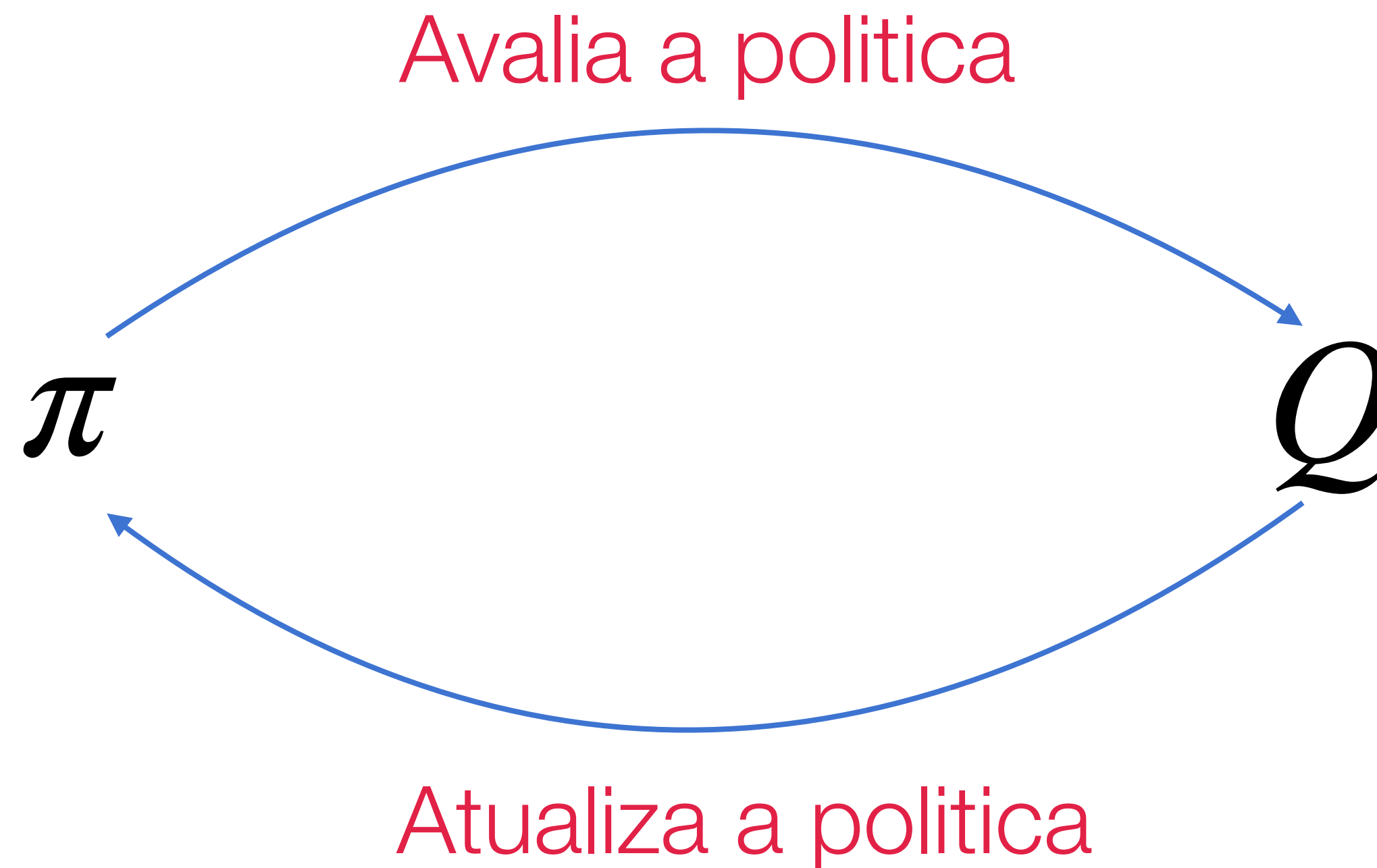
- De agora em diante, o MMC atualiza a politica π em função do maior valor $q(s, a)$

$$\pi'(s) = \operatorname{argmax}_a q_\pi(s, a)$$

- Em vez de um tabela com valores $V(s)$, o algoritmo constrói tabelas com valores $Q(s, a)$

Como um MMC resolve um problema de RL

- Diagrama de iteração de um algoritmo MMC



$$\pi_0 \rightarrow Q_{\pi_0} \rightarrow \pi_1 \rightarrow Q_{\pi_1} \rightarrow \dots \rightarrow Q_{\pi_\star} \rightarrow \pi_\star$$

Exploração de um algoritmo de MMC

Exploração e exploração

- As estimativas de $Q(s, a)$ melhorarão ao longo das iterações, porém Q pode não ser ótimo
- Por exemplo: uma ação a' pode ser ótima mas, como se tem uma má estimativa $Q(s, a')$, a' nunca será escolhido
- Precisamos assegurar que todas as ações sejam tentadas pelo menos uma vez

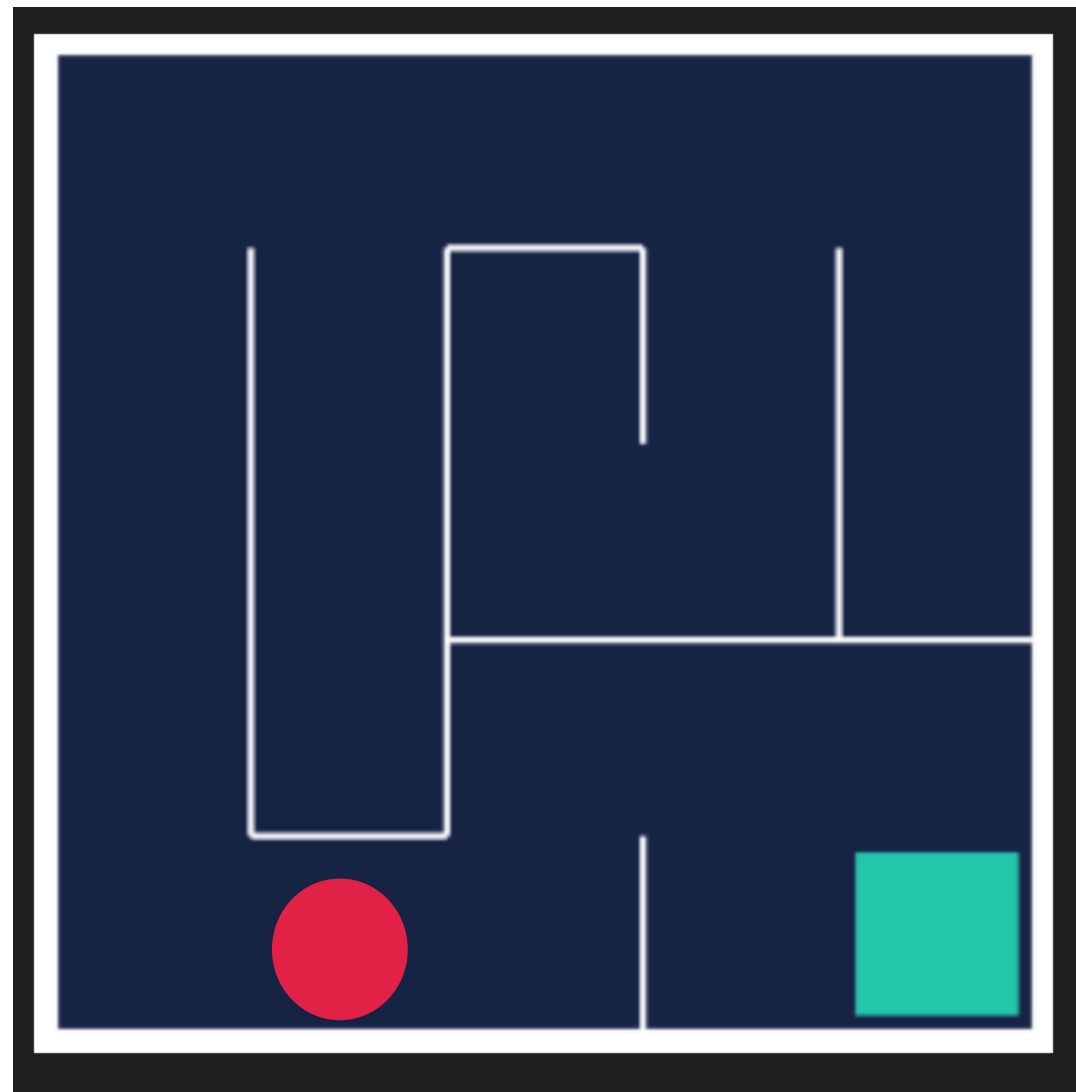
Exploração de um algoritmo de MMC

Exploração e exploração

- Como assegurar a exploração?

Exploração de estados iniciais

$$S_0 \sim S, \quad A_0 \sim A(S_0)$$



Políticas Estocásticas

$$\pi(a | s) > 0, \quad \forall a \in A(s)$$

Exploração de um algoritmo de MMC

Tipos de politicas estocasticas

- Dois tipos de politica estocástica

On-Policy

Obtém amostras usando a mesma politica π que será otimizada

Off-Policy

Utiliza uma politica exploratória π_b diferente da politica π que será otimizada

Aprendizado *On-Policy*

Política ϵ -gulosa

- Política ϵ -gulosa:

Escolhe-se uma ação aleatória
com probabilidade ϵ .

Escolhe-se a ação com maior
 $Q(s, a)$ com probabilidade $1 - \epsilon$.

$$\pi(a | s) = \begin{cases} 1 - \epsilon + \epsilon_r & a = a^\star \\ \epsilon_r & a \neq a^\star \end{cases}$$

$$\epsilon_r = \frac{\epsilon}{|A|}$$

Aprendizado *On-Policy*

Política ϵ -gulosa

- Política ϵ -gulosa, Exemplo:

Escolhe-se uma ação aleatória
com probabilidade ϵ .

Escolhe-se a ação com maior
 $Q(s, a)$ com probabilidade $1 - \epsilon$.

$$|A| = 4 \quad \epsilon = 0.2$$

$$\pi(a | s) = \begin{cases} 1 - 0.2 + 0.005 = 0.85 & a = a^* \\ 0.05 & a \neq a^* \end{cases} \quad \epsilon_r = \frac{0.2}{|4|} = 0.05$$

Aprendizado *On-Policy*

Algoritmo

Algorithm 1 On-policy Monte Carlo Control

```
1: Input:  $\epsilon$  random action probability,  $\gamma$  discount factor
2:  $\pi \leftarrow$  e-greedy policy w.r.t  $Q(s, a)$ 
3: Initialize  $Q(s, a)$  arbitrarily, with  $Q(\text{terminal}, \cdot) = 0$ 
4:  $G(s, a) \leftarrow []$ 
5: for episode  $\in 1..N$  do
6:   Generate episode following  $\pi : S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$ 
7:    $G \leftarrow 0$ 
8:   for  $t \in T - 1..0$  do
9:      $G \leftarrow R_{t+1} + \gamma G$ 
10:    Append  $G$  to  $G(S_t, A_t)$ 
11:     $Q(s, a) \leftarrow \text{average}(G(S_t, A_t))$ 
12:   end for
13: end for
14: Output: Near optimal policy  $\pi$  and action values  $Q(s, a)$ 
```

Aprendizado *On-Policy*

Abra o Notebook MC_OnPolicy.ipynb

Aprendizado *On-Policy* com α constante

Algoritmo

Algorithm 1 On-policy Monte Carlo Control

```
1: Input:  $\epsilon$  random action probability,  $\gamma$  discount factor
2:  $\pi \leftarrow$  e-greedy policy w.r.t  $Q(s, a)$ 
3: Initialize  $Q(s, a)$  arbitrarily, with  $Q(\text{terminal}, \cdot) = 0$ 
4:  $G(s, a) \leftarrow []$ 
5: for episode  $\in 1..N$  do
6:   Generate episode following  $\pi : S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$ 
7:    $G \leftarrow 0$ 
8:   for  $t \in T - 1..0$  do
9:      $G \leftarrow R_{t+1} + \gamma G$ 
10:    Append  $G$  to  $G(S_t, A_t)$ 
11:     $Q(s, a) \leftarrow \text{average}(G(S_t, A_t)) Q(s, a) + \alpha[G - Q(s, a)]$ 
12:   end for
13: end for
14: Output: Near optimal policy  $\pi$  and action values  $Q(s, a)$ 
```

Aprendizado *On-Policy* com α constante

Abra o Notebook MC_OnPolicyAC.ipynb

Aprendizado *Off-Policy*

Definição

- Em vez de se manter somente atualizando π por todo o tempo, usa-se ocasionalmente uma politica sub-ótima π_b para explorar estados não visitados

$$\pi_b(a | s)$$

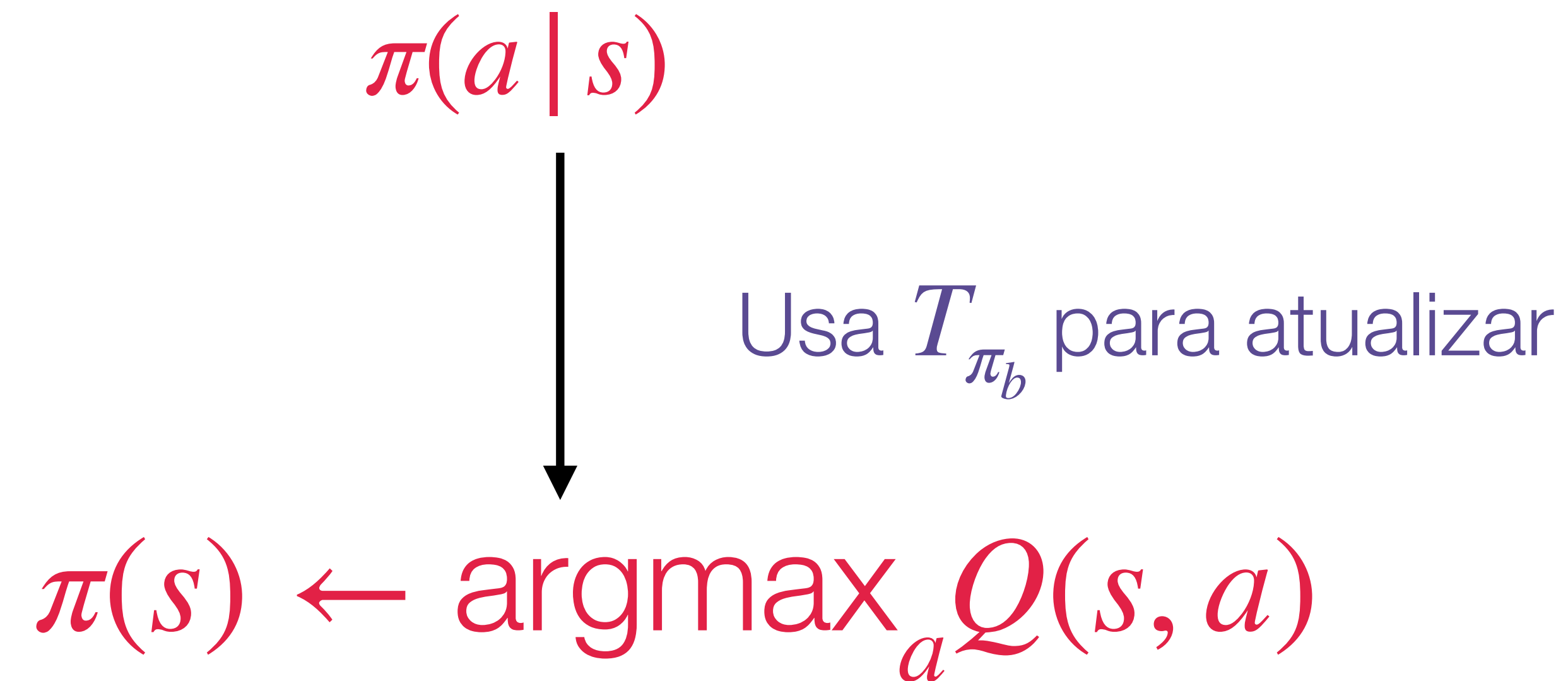
Gera trajetória T_{π_b}

$$T_{\pi_b} \leftarrow S_0, A_0, R_1, S_1, A_1, R_2, \dots, S_{T-1}, A_{T-1}, R_T$$

Aprendizado *Off-Policy*

Definição

- Em vez de se manter somente atualizando π por todo o tempo, usa-se ocasionalmente uma politica sub-ótima π_b para explorar estados não visitados



Aprendizado *Off-Policy*

Definição

- A politica π_b deve ser capaz de explorar todos os estados que π pode tomar:

se $\pi(a | s) > 0$, então $\pi_b(a | s) > 0$

π : politica alvo

π_b : politica exploratória

Aprendizado *Off-Policy*

Correção

- A média G dos retornos irá aproximar os retornos sob a política π_b , mas é necessário aproximar os valores de π , então o coeficiente W é introduzido a fim de corrigir o calculo dos retornos:

$$\mathbb{E}_b[G_t | S_t = s, A_t = a] = q_b(s, a)$$



Fator de correção W

$$\mathbb{E}[W_t G_t | S_t = s] = v_\pi(s)$$

Aprendizado *Off-Policy*

Correção

- W é calculado através de uma técnica estatística chamada de *importance sampling*

$$W_t = \prod_{k=t}^{T-1} \frac{\pi(A_k | S_k)}{\pi_b(A_k | S_k)}$$

Probabilidade de tomar a
ação a seguindo π

Probabilidade de tomar a
ação a seguindo π_b

Aprendizado *Off-Policy*

Atualização de valores- q

- Para realizar a atualização dos valores $Q(s, a)$, originalmente se calcula a média dos ganhos G

$$[G_1, G_2, G_3, \dots, G_N]$$

$$Q(s, a) \leftarrow \frac{1}{N} \sum_{k=1}^N G_k$$

Aprendizado *Off-Policy*

Atualização de valores- q

- Uma forma mais eficiente de se atualizar é atualizar a média de acordo com o deslocamento dos ganhos

$$Q(s, a) \leftarrow Q(s, a) + \frac{W}{C(s, a)} [G - Q(s, a)]$$

$$C(s, a) = \sum_{k=1}^N W_k$$

Aprendizado *Off-Policy*

Algoritmo

Algorithm 2 Off-policy Monte Carlo Control

```
1: Input:  $\gamma$  discount factor
2:  $\pi \leftarrow$  greedy policy w.r.t  $Q(s, a)$ 
3:  $b \leftarrow$  arbitrary policy with coverage of  $\pi$ 
4:  $C(s, a) \leftarrow 0$ 
5: Initialize  $Q(s, a)$  arbitrarily
6: for episode  $\in 1..N$  do
7:   Generate episode following  $b : S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$ 
8:    $G \leftarrow 0$ 
9:    $W \leftarrow 1$ 
10:  for  $t \in T - 1..0$  do
11:     $G \leftarrow R_{t+1} + \gamma G$ 
12:     $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$ 
13:     $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$ 
14:    if  $A_t \neq \pi(S_t)$  then
15:      Break the loop, move to next episode.
16:    end if
17:     $W \leftarrow W \frac{1}{b(A_t|S_t)}$ 
18:  end for
19: end for
20: Output: Optimal  $\pi$  and action values  $Q(s, a)$ 
```

Aprendizado *Off-Policy*

Abra o Notebook MC_OffPolicy.ipynb