Métodos de Diferença temporal (TD)

O que são

- Base para a maioria dos métodos avançados de RL:
  - Tentativa e erro do MMC: aprende por experiência
  - Amostragem da PD: estimativas de valores $Q(s, a)$
  - Memória

O que são

$$\begin{tikzcd}[column sep=huge]
\pi \arrow[r, bend left, "\text{Avalia a politica}"] & Q \arrow[l, bend left, "\text{Atualiza a politica}"]
\end{tikzcd}$$

O que são

Comparação

- Métodos de Monte-Carlo esperam ate o fim de um episódio para calcular G_t e atualizar $Q(s, a)$

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots + \gamma^{T-t-1} R_T$$

- Métodos de TD atualizam $Q(s, a)$ toda vez que um agente realiza uma ação

Alicerce de um Método de TD

- Como nos outros métodos, uma tabela de valores $q(s, a)$ é construída para cada par S_t, A_t

$$q_\pi(s, a) = \mathbb{E}[G_t \,|\, S_t = s, A_t = a] \qquad q_\pi(s, a) = \sum_{s', r} p(s', r \,|\, s, a)[r + \gamma v_\pi(s')]$$

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

Alicerce de um Método de TD

$$q_\pi(s,a) = \sum_{s',r} \underbrace{p(s', r \mid s, a)}_{R_{t+1} \qquad S_{t+1}} \left[r + \gamma \sum_{a'} \underbrace{\pi(a' \mid s')}_{A_{t+1}} \underbrace{q_\pi(s', a')}_{Q} \right]$$

Alicerce de um Método de TD

- $Q_\pi(S_t, A_t)$ podem ser estimados por:

$$R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})$$

$$\boxed{R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)}$$

Erro de Diferença temporal

Alicerce de um Método de TD

- Diferentes métodos possuem diferentes estimativas de $Q(S_t, A_t)$

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \boxed{[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]}$$

Lembre-se do Método de Monte-Carlo com α constante

Alicerce de um Método de TD

- Reorganizando:

$$Q(S_t, A_t) \leftarrow (1 - \alpha)Q(S_t, A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

SARSA

Primeiro algoritmo de TD

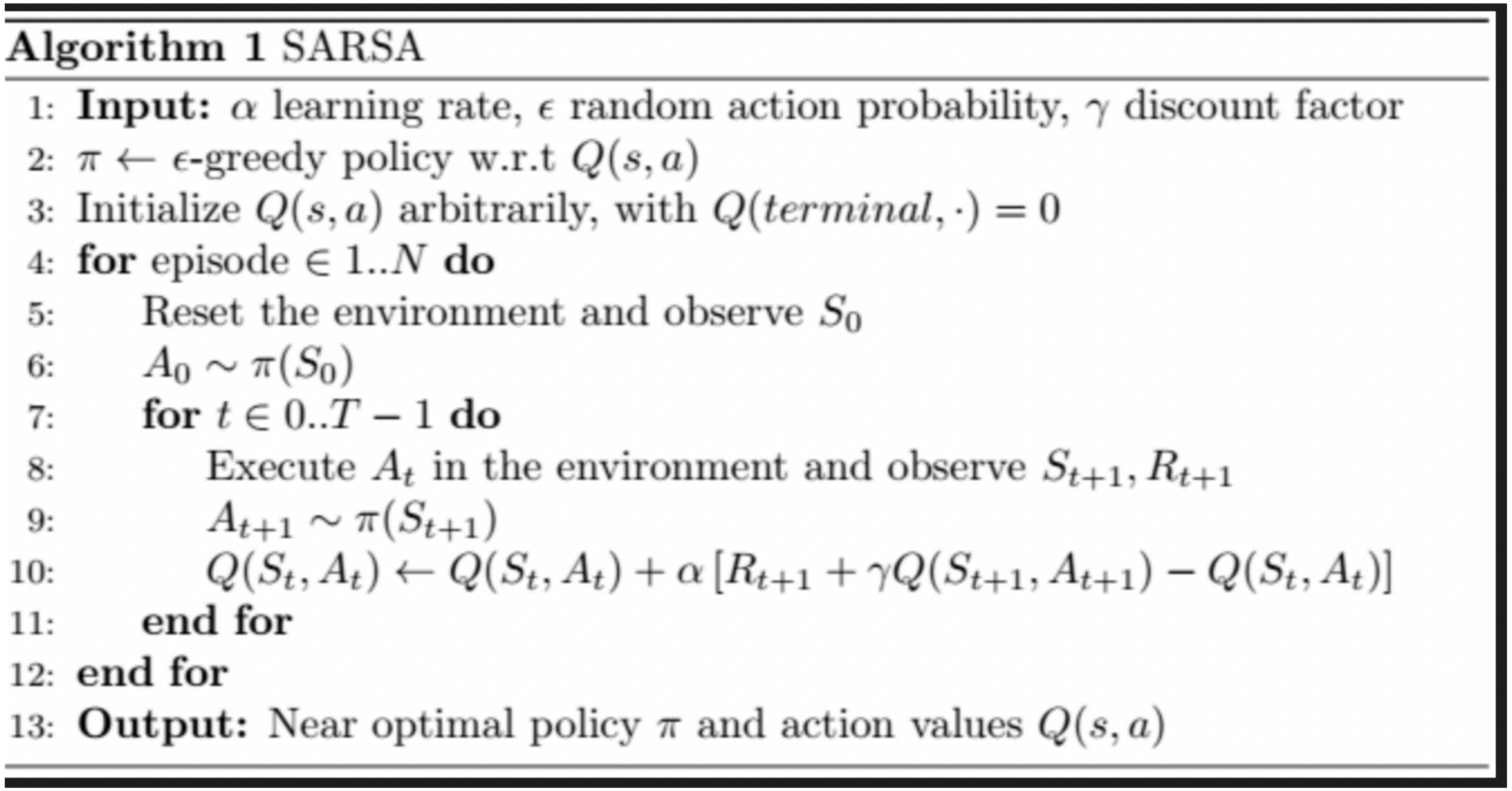
- SARSA: $S_t, A_t, R_t, S_{t+1}, A_{t+1}$

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

- O SARSA é *on-policy*, só tem uma politica
- Como no MMC, usa a politica ϵ-gulosa para exploração

SARSA

Primeiro algoritmo de TD



Algorithm 1 SARSA

```
1:  **Input:** $\alpha$ learning rate, $\epsilon$ random action probability, $\gamma$ discount factor
2:  $\pi \leftarrow \epsilon$-greedy policy w.r.t $Q(s, a)$
3:  Initialize $Q(s, a)$ arbitrarily, with $Q(terminal, \cdot) = 0$
4:  **for** episode $\in 1..N$ **do**
5:      Reset the environment and observe $S_0$
6:      $A_0 \sim \pi(S_0)$
7:      **for** $t \in 0..T - 1$ **do**
8:          Execute $A_t$ in the environment and observe $S_{t+1}, R_{t+1}$
9:          $A_{t+1} \sim \pi(S_{t+1})$
10:         $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\,[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$
11:     **end for**
12: **end for**
13: **Output:** Near optimal policy $\pi$ and action values $Q(s, a)$
```

SARSA

Primeiro algoritmo de TD

Abra o Notebook SARSA.ipynb

Q-Learning

Segundo Algoritmo de TD

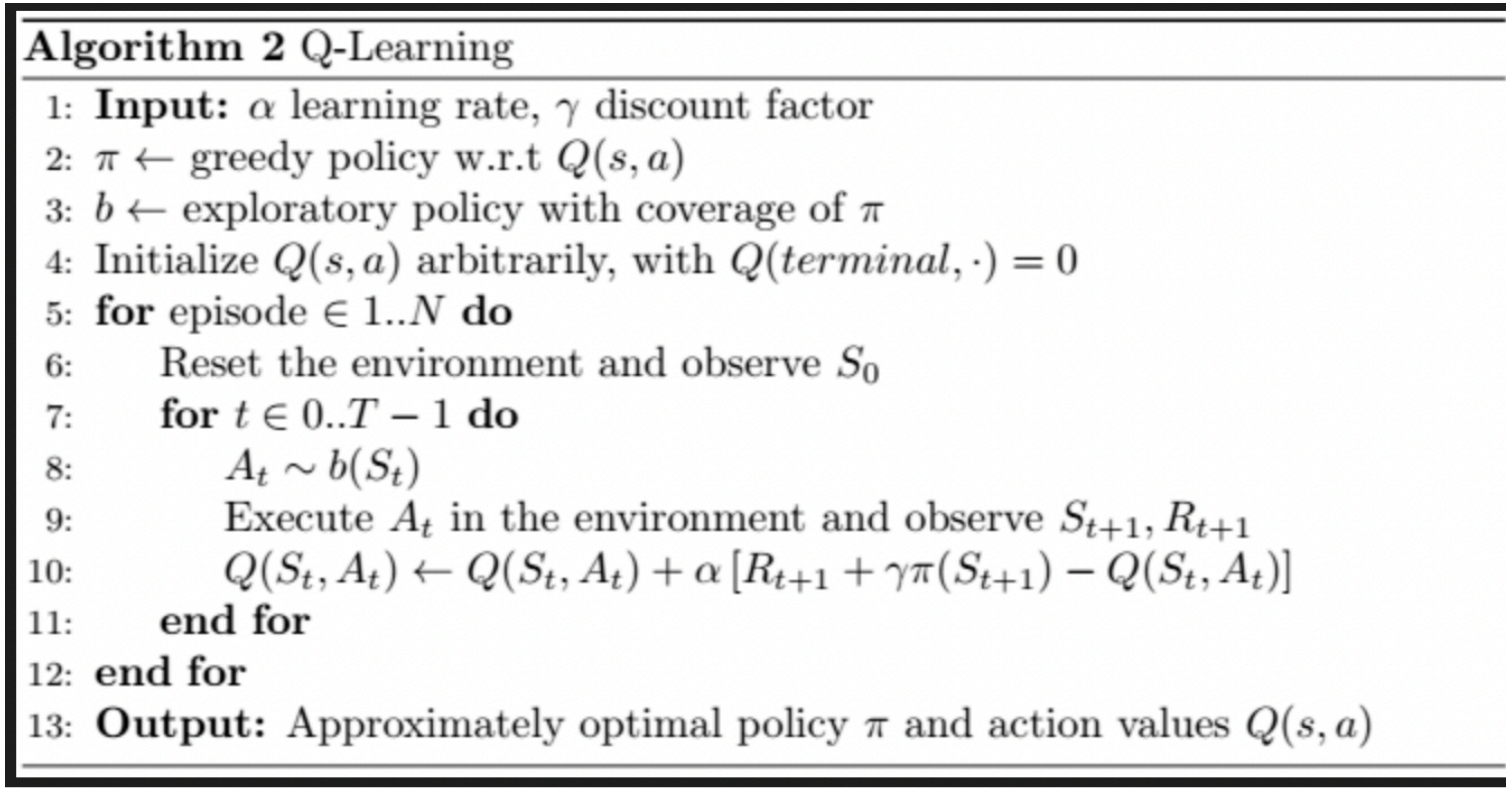
- O Q-Learning é *off-policy*, possui uma politica exploratória π_b
- Funciona exatamente como o SARSA, mas usa π_b para gerar uma trajetória

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

$$A_{t+1} = \pi(S_{t+1}) = \text{argmax}_a Q(S_{t+1}, a)$$

Q-Learning

Segundo Algoritmo de TD



Algorithm 2 Q-Learning

```
1:  Input: $\alpha$ learning rate, $\gamma$ discount factor
2:  $\pi \leftarrow$ greedy policy w.r.t $Q(s,a)$
3:  $b \leftarrow$ exploratory policy with coverage of $\pi$
4:  Initialize $Q(s,a)$ arbitrarily, with $Q(terminal, \cdot) = 0$
5:  for episode $\in 1..N$ do
6:      Reset the environment and observe $S_0$
7:      for $t \in 0..T-1$ do
8:          $A_t \sim b(S_t)$
9:          Execute $A_t$ in the environment and observe $S_{t+1}, R_{t+1}$
10:         $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\,[R_{t+1} + \gamma\pi(S_{t+1}) - Q(S_t, A_t)]$
11:     end for
12: end for
13: Output: Approximately optimal policy $\pi$ and action values $Q(s,a)$
```

Q-Learning

Segundo algoritmo de TD

Abra o Notebook QLearning.ipynb

Métodos de n-passos

Generalização de Métodos de TD

- Regra de atualização do SARSA

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

- *Bootstrapping* de 1 passo (1-*step*)

$$R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})$$

Métodos de $\boldsymbol{n}$-passos

Generalização de Métodos de TD

- *Bootstrapping* de 2 passos (2-*step*)

$$R_{t+1} + \gamma R_{t+2} + \gamma^2 Q(S_{t+2}, A_{t+2})$$

- *Bootstrapping* de 3 passos (3-*step*)

$$R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 Q(S_{t+3}, A_{t+3})$$

- *Bootstrapping* de n passos (n-*step*)

$$R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n Q(S_{t+n}, A_{t+n})$$

Retorno de n-passos

- Estimativa de n-retornos

$$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^{n} Q(S_{t+n}, A_{t+n})$$

SARSA *n*-passos

- SARSA é apenas um caso especial de *1-step bootstrapping*

$$G_{t:t+n} = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})$$

$$Q(S_t, A_t) \to Q(S_t, A_t) + \alpha[G_{t:t+n} - Q(S_t, A_t)$$

Retorno de n-passos

$$G_{t:t+2} = R_{t+1} + \gamma R_{t+2} + \gamma^2 Q(S_{t+2}, A_{t+2})$$

$$G_{t:t+3} = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 Q(S_{t+3}, A_{t+3})$$

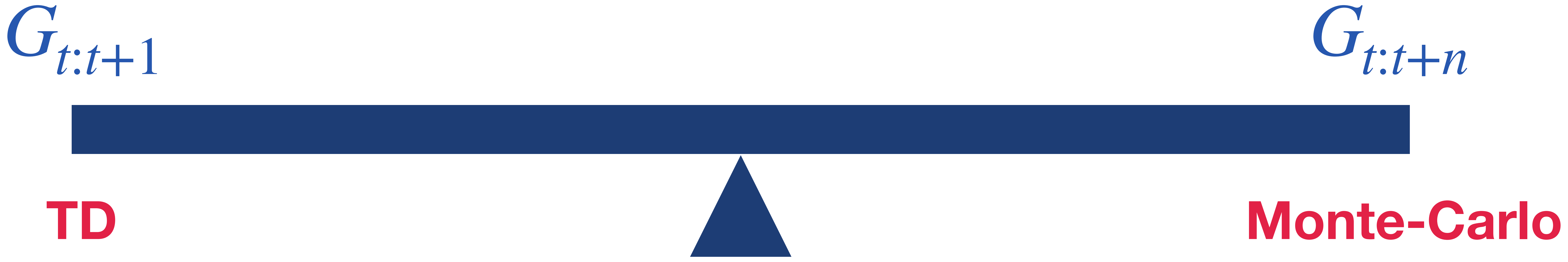
$$\cdots$$

Se $n \geq T$:

$$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n}$$ **Monte-Carlo**

$$Q(S_t, A_t) \rightarrow Q(S_t, A_t) + \alpha[G_{t:t+n} - Q(S_t, A_t)]$$

Generalização de algoritmos de $\boldsymbol{n}$-passos

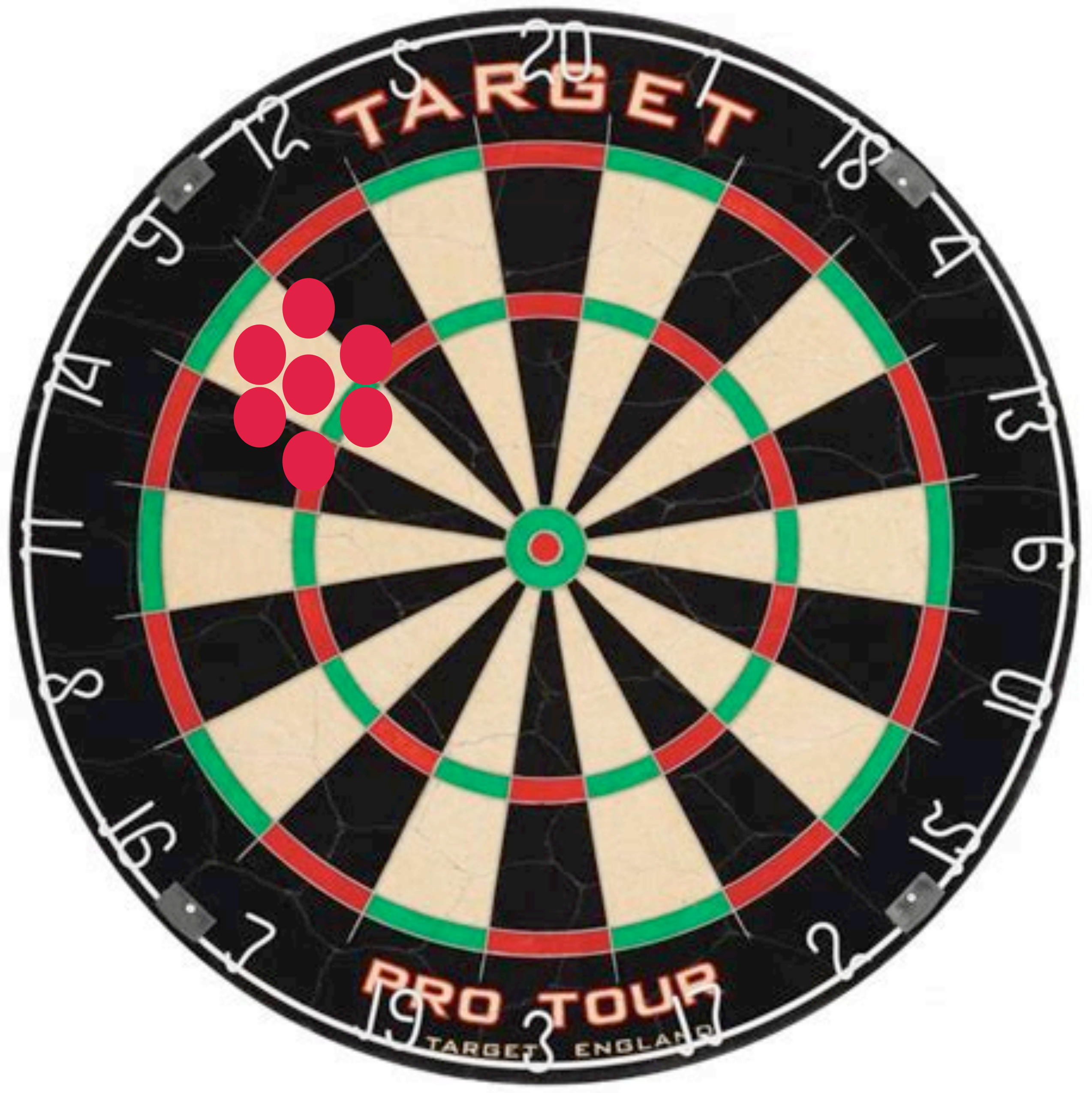


Generalização de algoritmos de n-passos

- Para que serve os valores de n?
- Regulagem do viés (*bias*) e da variância (*variance*)

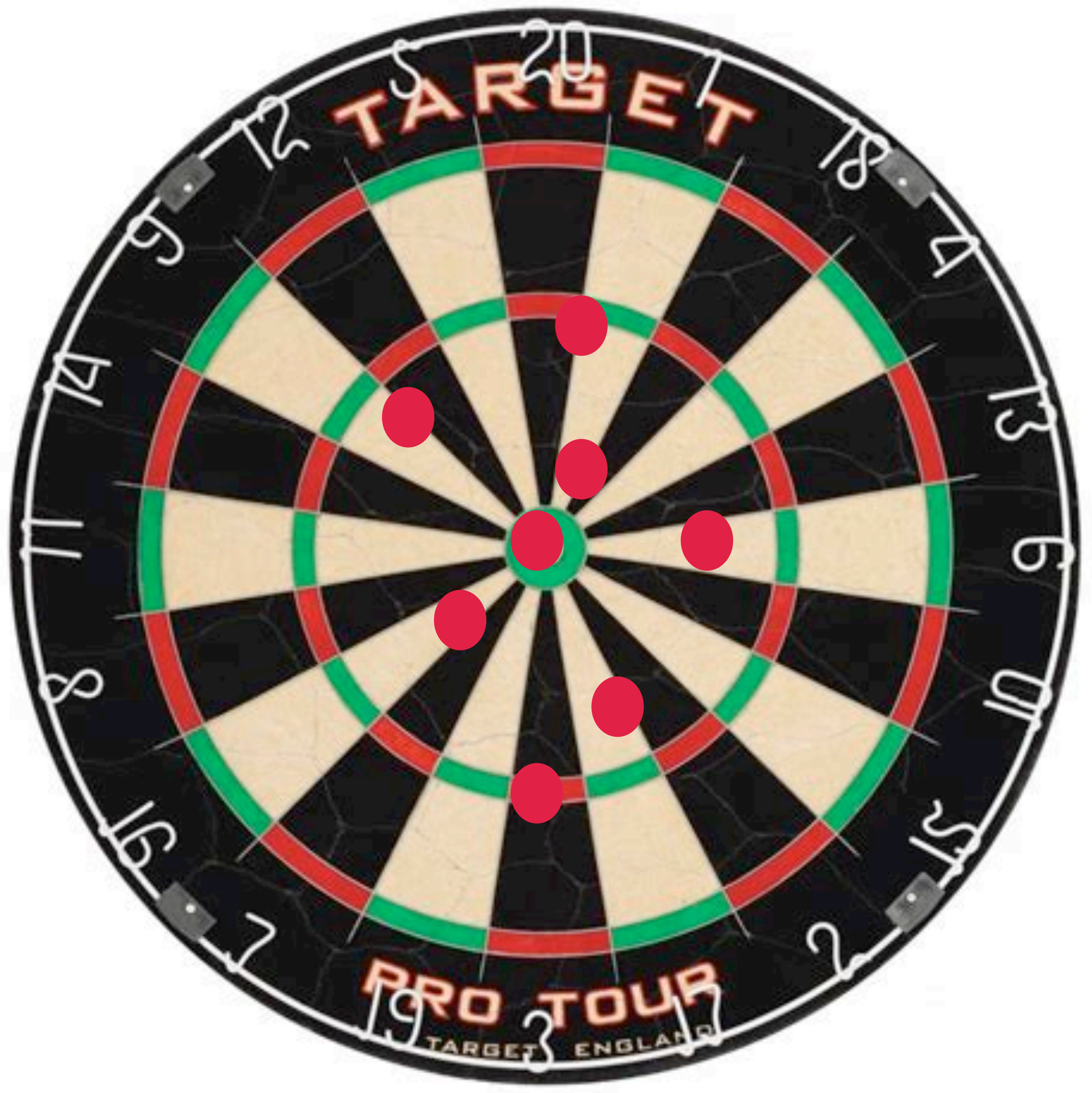
Generalização de algoritmos de n-passos

- Viés (*bias*)



Generalização de algoritmos de n-passos

- Variância (*variance*)



Generalização de algoritmos de n-passos

- Quanto maior n menor o viés
- Quanto maior n maior a variância

$$G_{t:t+2} = R_{t+1} + \gamma R_{t+2} + \gamma^2 Q(S_{t+2}, A_{t+2})$$

$$G_{t:t+3} = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 Q(S_{t+3}, A_{t+3})$$

$$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n}$$

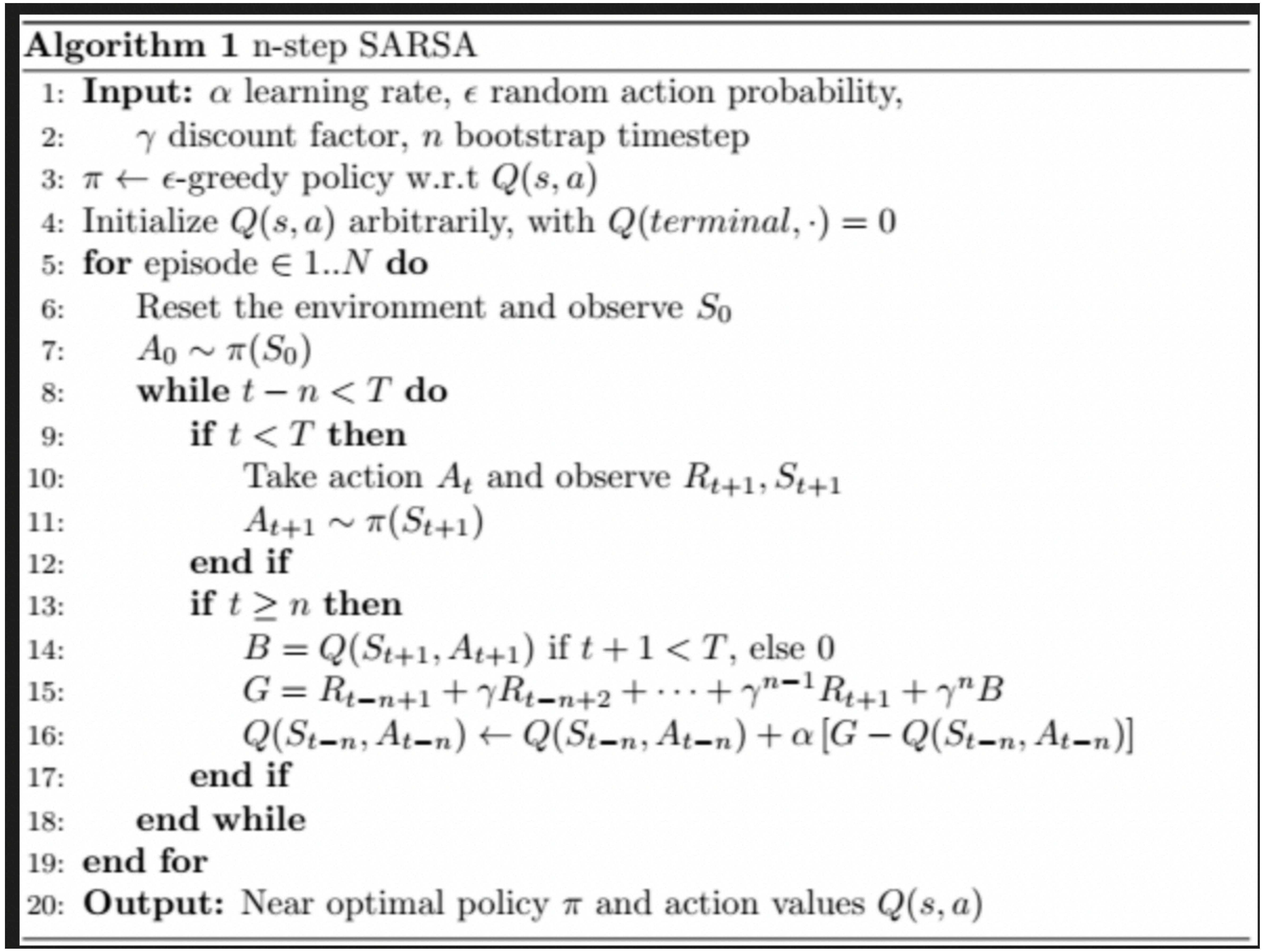
SARSA $\boldsymbol{n}$-passos

- Combina-se o SARSA com o n-*step bootstrapping*
- *On-Policy* com politica ϵ-gulosa

$$Q(S_t, A_t) \rightarrow Q(S_t, A_t) + \alpha[G_{t:t+n} - Q(S_t, A_t)]$$

$$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n Q(S_{t+1}, A_{t+1})$$

SARSA n-passos



Algorithm 1 n-step SARSA

```
1: Input: $\alpha$ learning rate, $\epsilon$ random action probability,
2:     $\gamma$ discount factor, $n$ bootstrap timestep
3: $\pi \leftarrow \epsilon$-greedy policy w.r.t $Q(s, a)$
4: Initialize $Q(s, a)$ arbitrarily, with $Q(terminal, \cdot) = 0$
5: for episode $\in 1..N$ do
6:     Reset the environment and observe $S_0$
7:     $A_0 \sim \pi(S_0)$
8:     while $t - n < T$ do
9:         if $t < T$ then
10:            Take action $A_t$ and observe $R_{t+1}, S_{t+1}$
11:            $A_{t+1} \sim \pi(S_{t+1})$
12:        end if
13:        if $t \geq n$ then
14:            $B = Q(S_{t+1}, A_{t+1})$ if $t + 1 < T$, else 0
15:            $G = R_{t-n+1} + \gamma R_{t-n+2} + \cdots + \gamma^{n-1} R_{t+1} + \gamma^n B$
16:            $Q(S_{t-n}, A_{t-n}) \leftarrow Q(S_{t-n}, A_{t-n}) + \alpha [G - Q(S_{t-n}, A_{t-n})]$
17:        end if
18:    end while
19: end for
20: Output: Near optimal policy $\pi$ and action values $Q(s, a)$
```

N-Step SARSA

Terceiro algoritmo de TD

Abra o Notebook NStepSARSA.ipynb