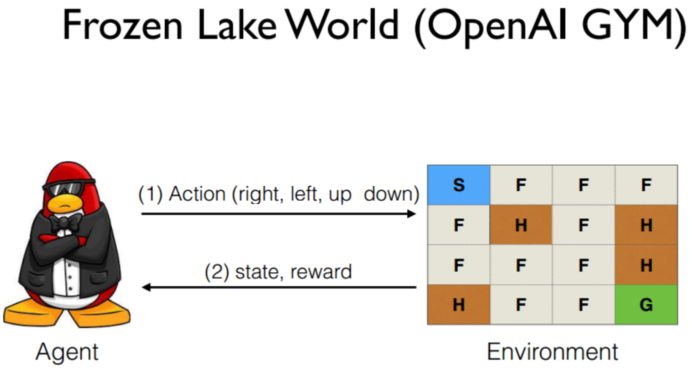
Lecture 4: Model-Free Reinforcement Learning: Monte-Carlo, SARSA, Q-Learning

Anton Plaksin

Пример: Frozen Lake



Example: Breakout Atari Game

- States: pixels
- Actions: $\rightarrow$, $\leftarrow$, «0»
- Rewards: points
- Final states: when the ball falls down

Markov Decision Process

Markov Property

$$\mathbb{P}[S_{t+1}|S_t, A_t] = \mathbb{P}[S_{t+1}|S_1, A_1, S_2, A_2 \ldots, S_t, A_t]$$

$$\mathbb{P}[R_t|S_t, A_t] = \mathbb{P}[R_t|S_1, A_1, S_2, A_2 \ldots, S_t, A_t] = 1$$

Markov Decision Process $\langle \mathcal{S}, \mathcal{S}_F, \mathcal{A}, \mathcal{P}, \mathcal{P}_0, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ is a finite $(|\mathcal{S}| = n)$ state space
- $\mathcal{S}_F$ is a set of final states
- $\mathcal{A}$ is a finite $(|\mathcal{A}| = m)$ action space
- $\mathcal{P}$ is an uknown transition probability function

$$\mathcal{P}(s'|s, a) = \mathbb{P}[S_{t+1} = s'|S_t = s, A_t = a]$$

- $\mathcal{P}_0$ is an uknown initial state probability function
- $\mathcal{R}$ is an uknown reward function

$$\mathcal{R}(s, a) = R_t \quad \Leftrightarrow \quad \mathbb{P}[R_t|S_t = s, A_t = a] = 1$$

- $\gamma \in [0, 1]$ is a discount coefficient

Model-Free Algorithms

- Monte-Carlo Algorithm
- SARSA Algorithm
- Q-Learning Algorithm

Policy Iteration

Let π^0 and $L, K \in \mathbb{N}$

- For each $k \in \overline{0, K}$, do

  (Policy evaluation) Iterative Policy Evaluation:

$$v^{l+1}(s) = \sum_a \pi(a|s)\Big(\mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s'|s,a) v^l(s')\Big),\ l \in \overline{0, L-1}.$$

- Define $q^L(s,a)$ by $v^L(s)$

  (Policy improvement) Greedy Policy Improvement:

$$\pi^{k+1}(a|s) = \begin{cases} 1, & \text{if } a \in \operatorname{argmax}_{a' \in \mathcal{A}} q^L(s,a') \\ 0, & \text{otherwise} \end{cases}$$

$$q^L(s,a) = \mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s'|s,a) v^L(s')$$

Поскольку мы не знаем функцию наград и вероятности перехода в следующее состояние в данной интерпретации посчитать q-функцию невозможно. Воспользуемся методом Монте-Карло.

Trajectory. Deterministic Case

- Set π
- Agent starts from an initial state S_0
- acts $A_0 = \pi(S_0)$ $\quad q_\pi(S_0, A_0) = G_0$
- gets a reward R_0 and goes to a next state S_1
- acts $A_1 = \pi(S_1)$ $\quad q_\pi(S_1, A_1) = G_1$
- $\dots$
- acts $A_{T-2} = \pi(S_{T-2})$, $\quad q_\pi(S_{T-2}, A_{T-2}) = G_{T-2}$
- gets a reward R_{T-2} and goes to a next state S_{T-1}
- acts $A_{T-1} = \pi(S_{T-1})$, $\quad q_\pi(S_{T-1}, A_{T-1}) = G_{T-1}$
- gets a reward R_{T-1} and goes to a next state $S_T \in \mathcal{S}_F$
- $\tau = \{S_0, A_0, S_1, A_1, \dots, S_T\}, \quad G(\tau) = \sum_{t=0}^{T-1} \gamma^t R_t, \quad G_t = \sum_{k=t}^{T-1} \gamma^{k-t} R_t$

Problem

$$\text{to find} \quad q_\pi(s, a) = \mathbb{E}_\pi[G \,|\, S_0 = s, A_0 = a]$$

Trajectory. General Case

- Set $\pi(a|s)$. Initialize $W(s,a) = 0$ and $N(s,a) = 0$
- Agent starts from an initial state S_0,
- acts $A_0 \sim \pi(\cdot|S_0)$
  $W(S_0, A_0) \leftarrow W(S_0, A_0) + G_0,\ \ N(S_0, A_0) \leftarrow N(S_0, A_0) + 1$
  $Q(S_0, A_0) \leftarrow W(S_0, A_0)/N(S_0, A_0)$
- gets a reward R_0 and goes to a next state S_1
- acts $A_1 \sim \pi(\cdot|S_1)$
  $W(S_1, A_1) \leftarrow W(S_1, A_1) + G_1,\ \ N(S_1, A_1) \leftarrow N(S_1, A_1) + 1$
  $Q(S_1, A_1) \leftarrow W(S_1, A_1)/N(S_1, A_1)$
- $\dots$
- acts $A_{T-1} \sim \pi(\cdot|S_{T-1})$, $W(S_{T-1}, A_{T-1}) \leftarrow W(S_{T-1}, A_{T-1}) + G_{T-1}$,
  $N(S_{T-1}, A_{T-1}) \leftarrow N(S_{T-1}, A_{T-1}) + 1$
  $Q(S_{T-1}, A_{T-1}) \leftarrow W(S_{T-1}, A_{T-1})/N(S_{T-1}, A_{T-1})$
- gets a reward R_{T-1} and goes to a next state $S_T \in \mathcal{S}_F$
- $\tau = \{S_0, A_0, S_1, A_1, \dots, S_T\}$, $\quad G(\tau) = \sum\limits_{t=0}^{T-1} \gamma^t R_t$, $\quad G_t = \sum\limits_{k=t}^{T-1} \gamma^{k-t} R_t$

Problem

$$\text{to find} \quad q_\pi(s,a) = \mathbb{E}_\pi[G \,|\, S_0 = s, A_0 = a] \approx Q(s,a)$$

Recurrent Formula

$$Q_N = \frac{1}{N}\sum_{i=1}^{N} w_i$$

Then

$$Q_{N+1} = \frac{1}{N+1}\sum_{i=1}^{N+1} w_i = \frac{1}{N+1}\left(\sum_{i=1}^{N} w_i + w_{N+1}\right)$$

$$= \frac{1}{N+1}(NQ_N + w_{N+1}) = Q_N + \frac{1}{N+1}(w_{N+1} - Q_N)$$

$$Q_{N+1} = Q_N + \frac{1}{N+1}(w_{N+1} - Q_N)$$

Trajectory. General Case

- Set $\pi(a|s)$. Initialize $Q(s,a) = 0$ and $N(s,a) = 0$
- Agent starts from an initial state S_0,
- acts $A_0 \sim \pi(\cdot|S_0)$ $Q(S_0, A_0) \leftarrow Q(S_0, A_0) + \frac{1}{N(S_0,A_0)+1}(G_0 - Q(S_0, A_0))$, $N(S_0, A_0) \leftarrow N(S_0, A_0) + 1$
- gets a reward R_0 and goes to a next state S_1
- acts $A_1 \sim \pi(\cdot|S_1)$ $Q(S_1, A_1) \leftarrow Q(S_1, A_1) + \frac{1}{N(S_1,A_1)+1}(G_1 - Q(S_1, A_1))$, $N(S_1, A_1) \leftarrow N(S_1, A_1) + 1$
- ...
- acts $A_{T-1} \sim \pi(\cdot|S_{T-1})$, $Q(S_{T-1}, A_{T-1}) \leftarrow Q(S_{T-1}, A_{T-1}) + \frac{1}{N(S_{T-1},A_{T-1})+1}(G_{T-1} - Q(S_{T-1}, A_{T-1}))$, $N(S_{T-1}, A_{T-1}) \leftarrow N(S_{T-1}, A_{T-1}) + 1$
- gets a reward R_{T-1} and goes to a next state S_T
- $\tau = \{S_0, A_0, S_1, A_1, \ldots, S_T\}$, $G(\tau) = \sum_{t=0}^{T-1} \gamma^t R_t$, $G_t = \sum_{k=t}^{T-1} \gamma^{k-t} R_t$

Problem

$$\text{to find} \quad q_\pi(s,a) = \mathbb{E}_\pi[G \mid S_0 = s, A_0 = a] \approx Q(s,a)$$

Monte-Carlo Policy Evaluation

Let π be fixed. Set $Q(s,a) = 0$ and $N(s,a) = 0$.
For each $k \in \overline{1, K}$, do

- According to π, get a trajectory $\tau = (S_0, A_0, \ldots, S_T)$ and rewards $(R_0, \ldots, R_{T-1})$. Define $(G_0, \ldots, G_{T-1})$.

  For each $t \in \overline{0, T-1}$, update Q and N:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t) + 1}\big(G_t - Q(S_t, A_t)\big),$$

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

$$Q(s,a) \approx q_\pi(s,a)$$

Will it work?

Let π^0 be initialized and $K > 0$.
For each $k \in \overline{1, K}$, do

- (Policy evaluation) Monte-Carlo Policy Evaluation. Obtain $Q^k(s,a) \approx q_{\pi^k}(s,a)$

  (Policy improvement) Greedy Policy Improvement:

$$\pi^{k+1}(a|s) = \begin{cases} 1, & \text{if } a \in \operatorname{argmax}_{a' \in \mathcal{A}} Q^k(s,a') \\ 0, & \text{otherwise} \end{cases}$$

Данный подход будет работать плохо, поскольку получая детерминированную политику мы теряем возможность исследования (особенно плохо при плохой инициализации политики или при маленьком количестве итераций).

ε-Greedy Policy Improvement

$\pi = \varepsilon\text{-greedy}(Q)$

$$\pi(a|s) = \begin{cases} 1 - \varepsilon + \varepsilon/m, & \text{if } a \in \text{argmax}_{a' \in \mathcal{A}} Q(s, a'), \\ \varepsilon/m, & \text{otherwise} \end{cases}$$

Policy Improvement Theorem

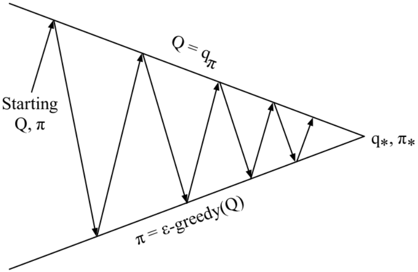
Let $Q(s, a)$ be defined.
Let $\pi = \varepsilon\text{-greedy}(Q)$ and $\pi' = \varepsilon\text{-greedy}(q_\pi)$.
Then $\pi' \geq \pi$ $(v_{\pi'}(s) \geq v_\pi(s), \forall s)$

Эпсилон-жадный подход к улучшению политики решает эту проблему.

Learning with Monte-Carlo Policy Evaluation

Let π^0 be initialized and $K > 0$, $\varepsilon = 1$.
For each $k \in \overline{1, K}$,do

- (Policy evaluation) Monte-Carlo Policy Evaluation - obtain $Q^k(s,a) \approx q_{\pi^k}(s,a)$
- (Policy improvement) ε-Greedy Policy Improvement - obtain π^{k+1} by Q^k. Define $\varepsilon = 1/k$



Theorem

$Q^k \to q_* $ and $\pi^k \to \pi_*$ as $k \to \infty$.

Monte-Carlo Algorithm

Let $Q(s,a) = 0$, $N(s,a) = 0$ and $\varepsilon = 1$.
For each $k \in \overline{1,K}$, do

- According to $\pi = \varepsilon\text{-greedy}(Q)$, get trajectory $\tau = (S_0, A_0, \dots, S_T)$ and rewards $(R_0, \dots, R_{T-1})$. Define $(G_0, \dots, G_{T-1})$.

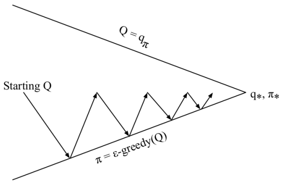
  For each $t \in \overline{0, T-1}$, update Q and N:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t) + 1}\big(G_t - Q(S_t, A_t)\big),$$

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

  Define $\varepsilon = 1/k$

Для сходимости алгоритма на каждом шаге достаточно использовать только одну траекторию!!!



Теорема

$Q^k \to q_*$ and $\pi^k \to \pi_*$ as $k \to \infty$.

Using Bellman Equation

Bellman Expectation Equation для q_π

$$q_\pi(s,a) = \mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s'|s,a) \sum_{a'} \pi(a'|s') q_\pi(s',a')$$

$$\Downarrow$$

$$q_\pi(s,a) = \mathbb{E}_\pi[R_t + \gamma q_\pi(S_{t+1}, A_{t+1}) \,|\, S_t = s,\ A_t = a]$$

$$\Downarrow$$

Temporal-Difference

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

Temporal-Difference Policy Evaluation

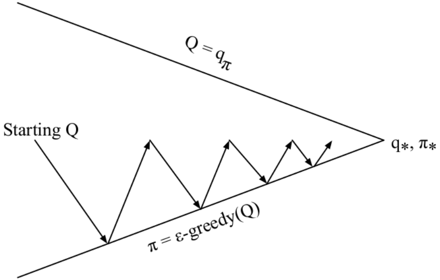
- Let π be fixed and $Q(s,a) = 0$.
  For each $k \in \overline{1, K}$, do
  - According to π, get trajectory $\tau = (S_0, A_0, \dots, S_T)$ and rewards $(R_0, \dots, R_{T-1})$.
    For each $t \in \overline{0, T-1}$, update Q:
    $$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))$$

$$Q(s,a) \approx q_\pi(s,a)$$

Другой подход к получению оценки Q:
расчет оценки через уравнение Беллмана по формуле

Learning with Temporal-Difference Policy Evaluation

- Let $Q(s,a) = 0$, $K > 0$, and $\varepsilon = 1$.
  For each $k \in \overline{1, K}$, do
  - According to $\pi = \varepsilon\text{-greedy}(Q)$, get trajectory $\tau = (S_0, A_0, \ldots, S_T)$ and rewards $(R_0, \ldots, R_{T-1})$.
    For each $t \in \overline{0, T-1}$, update Q:
    $$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))$$
    Define $\varepsilon = 1/k$



SARSA Algorithm

Let $Q(s,a) = 0$, $K > 0$, and $\varepsilon = 1$.

For each $k \in \overline{1, K}$, do

- During trajectory
  - From the state S_t, acting $A_t \sim \pi(\cdot|S_t)$, where $\pi = \varepsilon\text{-greedy}(Q)$, get R_t, go to the next state S_{t+1}, and act $A_{t+1} \sim \pi(\cdot|S_{t+1})$
  - According to $(S_t, A_t, R_t, S_{t+1}, A_{t+1})$, update Q:

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

- Put $\varepsilon = 1/k$

Theorem

$Q^k \to q_*$ and $\pi^k \to \pi_*$ as $k \to \infty$.

Using Bellman Optimality Equation

Bellman Optimality Equation для q_*

$$q_*(s,a) = \mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s'|s,a) \max_{a'} q_*(s',a')$$

$$\Downarrow$$

$$q_*(s,a) = \mathbb{E}[R_t + \gamma \max_{a'} q_*(S_{t+1},a') \,|\, S_t = s,\ A_t = a]$$

$$\Downarrow$$

Q-Learning

$$Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha(R_t + \gamma \max_{a'} Q(S_{t+1},a') - Q(S_t,A_t))$$

Q-Learning Algorithm

Let $Q(s,a) = 0$, $K > 0$, and $\varepsilon = 1$.

- For each $k \in \overline{1, K}$, do

  During trajectory

  - From the state S_t, acting $A_t \sim \pi(\cdot|S_t)$, where $\pi = \varepsilon$-greedy(Q), get R_t, and go to the next state S_{t+1}

    According to (S_t, A_t, R_t, S_{t+1}), update Q:

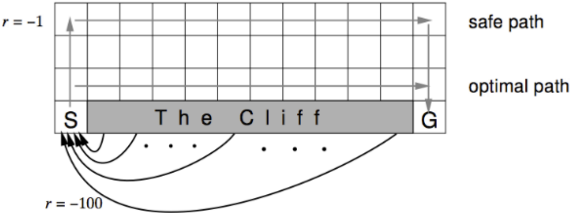
$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_t + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t))$$

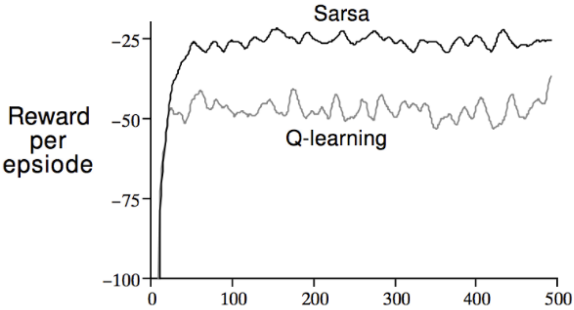
  Put $\varepsilon = 1/k$

Theorem

$Q^k \to q_*$ and $\pi^k \to \pi_*$ as $k \to \infty$.

Comparison of SARSA и Q-Learning





Dynamic Programming and Reinforcement Learning

Q-Policy Iteration	Sarsa
$Q(s,a) \leftarrow \mathbb{E}\left[R + \gamma Q(S',A') \mid s,a\right]$	$Q(S,A) \overset{\alpha}{\leftarrow} R + \gamma Q(S',A')$
Q-Value Iteration	**Q-Learning**
$Q(s,a) \leftarrow \mathbb{E}\left[R + \gamma \max_{a' \in \mathcal{A}} Q(S',a') \mid s,a\right]$	$Q(S,A) \overset{\alpha}{\leftarrow} R + \gamma \max_{a' \in \mathcal{A}} Q(S',a')$