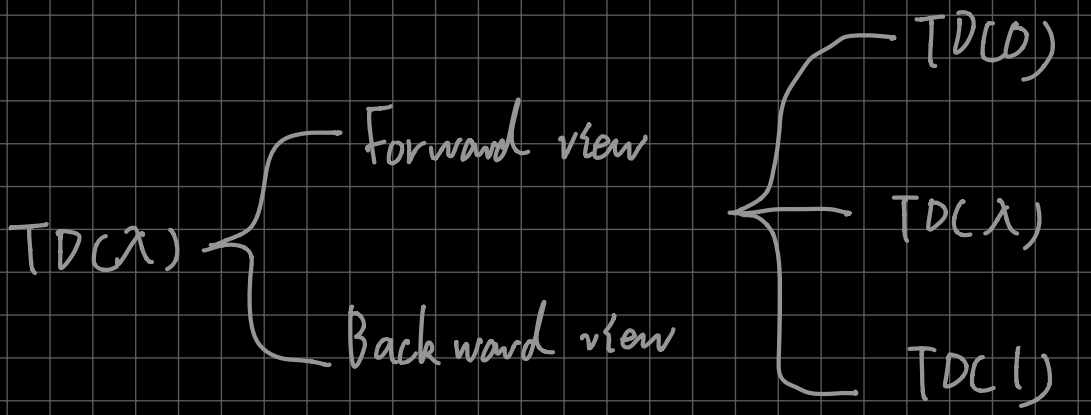
Lecture 5 TD(λ)



Forward view TD(λ):

$$V(S_t) \leftarrow V(S_t) + \alpha\left(G_t^{\lambda} - V(S_t)\right)$$

G_t^{λ}: λ return

$$G_t^{\lambda} = (1-\lambda)\cdot\sum_{n=1}^{\infty}\lambda^{n-1}\,G_t^{(n)}$$

$G_t^{(n)}$: n step return

注意此二者的书写：G_t^{λ}, $G_t^{(n)}$

$$G_t^{(n)} = \underbrace{R_{t+1} + \gamma\cdot R_{t+2} \cdots + \gamma^{n-1}\cdot R_{t+n}}_{n\text{项}} + V(S_{t+n})$$

$n\to\infty$

MC: $G_t^{\infty} = R_{t+1} + \gamma\cdot R_{t+2} \cdots + \gamma^{T-1}\cdot R_T$

R_T: terminal

Forward: $S_t \longrightarrow S_{t+1} \longrightarrow S_{t+2} \cdots\cdots \longrightarrow S_T$

Backward view TD(λ):

$$V(S_t) \leftarrow V(S_t) + \alpha \cdot \delta_t \cdot E_t(s)$$

(小写)

① TD error ② Eligibility trace

① $\delta_t = R_{t+1} + \gamma \cdot V(S_{t+1}) - V(S_t)$

(大写)

② $\begin{cases} E_t(s) = \gamma \cdot \lambda \cdot E_{t-1}(s) + \mathbb{1}(S_t = s) \\ E_0(s) = 0 \end{cases}$

Recency Heuristic; Frequency heuristic

Backward: $S_0 \cdots \leftarrow S_t \leftarrow S_{t+1}$

Summary of Forward view TD(λ) and Backward view TD(λ)

Offline updates	$\lambda = 0$	$\lambda \in (0,1)$	$\lambda = 1$
Backward view	TD(0)	TD(λ)	TD(1)
	$\Vert$	$\Vert$	$\Vert$
Forward view	TD(0)	Forward TD(λ)	MC
Online updates	$\lambda = 0$	$\lambda \in (0,1)$	$\lambda = 1$
Backward view	TD(0)	TD(λ)	TD(1)
	$\Vert$	$\nparallel$	$\nparallel$
Forward view	TD(0)	Forward TD(λ)	MC
	$\Vert$	$\Vert$	$\Vert$
Exact Online	TD(0)	Exact Online TD(λ)	Exact Online TD(1)

① In offline setting, Backward view and Forward view are equal.

$$MC = TD(\lambda)$$

Backward view　Forward View

② In online setting, Backward view and Forward view are no equal (except TD(0), obviously).

Forward TD(1) = Exact Online TD(1) = MC

Forward TD(λ) = Exact Online TD(λ)

Forward means "online"

Lecture 6 MDP Control

MDP Control problem
- On-policy
  - MC (TD(1))
  - TD
    - TD(0) ⇒ SARSA
    - TD(λ) ⇒ SARSA(λ) 狭义
- Off-policy ⇒ Q-learning

⇓ optimize the value function of unknown MDP

MDP Prediction problem:

(lecture 4, 5) ⇒ estimate the value function of unknown MDP ⇒ 广义 TD(λ)

(lecture 3) ⇒ estimate & optimize the value function of known MDP

⇓ policy evaluation ⇓ policy improvement

MC:

On-policy first-visit MC control (for ε-soft policies), estimates $\pi \approx \pi_*$

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
  $Q(s,a) \leftarrow$ arbitrary
  $Returns(s,a) \leftarrow$ empty list
  $\pi(a|s) \leftarrow$ an arbitrary ε-soft policy

Repeat forever:
  (a) Generate an episode using π
  (b) For each pair s, a appearing in the episode:
    policy evaluation: $G \leftarrow$ return following the first occurrence of s, a
    Append G to $Returns(s,a)$
    $Q(s,a) \leftarrow$ average$(Returns(s,a))$
  (c) For each s in the episode:
    policy improvement: $A^* \leftarrow \arg\max_a Q(s,a)$
    For all $a \in \mathcal{A}(s)$:

$$\pi(a|s) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(s)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(s)| & \text{if } a \neq A^* \end{cases}$$

why use ε-greedy, not use greedy?

Greedy strategy is model-free, so in on-policy situation can't ensure the best result. ε-greedy is the simplest way to ensure continual exploration.

⇒ ε-greedy, with ε prob randomly choose an action, 1-ε prob greedy choose.

SARSA:

Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Repeat (for each step of episode):
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma Q(S',A') - Q(S,A)\right]$
    $S \leftarrow S'$; $A \leftarrow A'$; ⇒ on-policy
  until S is terminal

Here we can directly choose from V-table, or in-directly choose from π-table.

But for latter option, we need to perform policy improvement for π-table. can be skip if we use former method.

SARSA(λ):

Initialize $Q(s,a)$ arbitrarily, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
Repeat (for each episode):
  $E(s,a) = 0$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$ ⇒ init the eligibility trace
  Initialize S, A ⇒ compare to SARSA, which use ε-greedy choose A. But in SARSA(λ), each episode reaches the end, so we just need to init A, not choose A.
  Repeat (for each step of episode):
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $\delta \leftarrow R + \gamma Q(S',A') - Q(S,A)$
    $E(S,A) \leftarrow E(S,A) + 1$
    For all $s \in \mathcal{S}, a \in \mathcal{A}(s)$: ⇒ for each step of episode, update all the state,
      $Q(s,a) \leftarrow Q(s,a) + \alpha\delta E(s,a)$ action item in Q-table to get λ-returns.
      $E(s,a) \leftarrow \gamma\lambda E(s,a)$
    $S \leftarrow S'$; $A \leftarrow A'$ ⇒ on-policy, go to next step.
  until S is terminal

Forward: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \cdot (q_t^{\lambda} - Q(S_t, A_t))$

q_t^{λ}: λ returns

$q_t^{\lambda} = (1-\lambda) \cdot \sum_{n=1}^{\infty} \lambda^{(n-1)} \cdot q_t^{(n)}$

$q_t^{(n)}$: n step Q returns

$q_t^{(n)} = R_{t+1} + \gamma \cdot R_{t+2} \cdots + \gamma^{n-1} \cdot R_{t+n} + Q(S_{t+n})$

Backward: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \cdot \delta_t \cdot E_t(s, a)$

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

$E_t(s, a) = \gamma \cdot \lambda \cdot E_{t-1}(s, a) + \mathbb{1}(S_t = s, A_t = a)$

Q-learning:

Initialize $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
    $S \leftarrow S'$;
  until S is terminal

behavior policy (μ) is ε-greedy

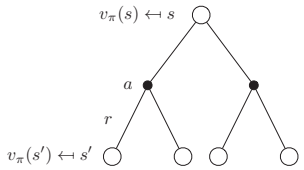
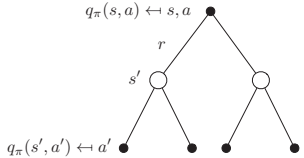
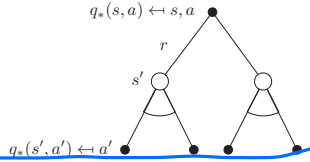
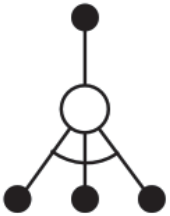
$\Rightarrow$ off-policy

target policy (π) is greedy

On-policy Vs Off-policy:

- On policy is generally simple.
- Off policy is more powerful and general.
  But it has greater variance and slower convergence.

Relationship Between DP and TD

	Full Backup (DP)	*Sample Backup (TD)*
Bellman Expectation Equation for $v_\pi(s)$	 Iterative Policy Evaluation	 TD Learning
Bellman Expectation Equation for $q_\pi(s,a)$	 Q-Policy Iteration	 Sarsa
Bellman Optimality Equation for $q_*(s,a)$	 Q-Value Iteration	 Q-Learning

predict problem

control problem

(TD(0), TD(λ), MC)

value iteration = policy iteration + policy improvement

etwork

Relationship Between DP and TD (2)

Predict Problem

Full Backup (DP)	*Sample Backup (TD)*
Iterative Policy Evaluation $V(s) \leftarrow \mathbb{E}\left[R + \gamma V(S') \mid s\right]$	TD Learning $V(S) \overset{\alpha}{\leftarrow} R + \gamma V(S')$ ⇒ TD(λ)
Q-Policy Iteration $Q(s,a) \leftarrow \mathbb{E}\left[R + \gamma Q(S',A') \mid s,a\right]$	Sarsa $Q(S,A) \overset{\alpha}{\leftarrow} R + \gamma Q(S',A')$ ⇒ SARSA(λ)
Q-Value Iteration (没有a即 Value Iteration) $Q(s,a) \leftarrow \mathbb{E}\left[R + \gamma \max_{a' \in \mathcal{A}} Q(S',a') \mid s,a\right]$	Q-Learning $Q(S,A) \overset{\alpha}{\leftarrow} R + \gamma \max_{a' \in \mathcal{A}} Q(S',a')$

Control problems

where $x \overset{\alpha}{\leftarrow} y \equiv x \leftarrow x + \alpha(y - x)$

Institute of Media, Information, and Network