

# Lecture 8: Model-Free RL Part II

Ziyu Shao

School of Information Science and Technology  
ShanghaiTech University

April 23, 2025

# Outline

- 1 Introduction
- 2 On-Policy Monte-Carlo Control
- 3 On-Policy Temporal-Difference Learning
- 4 Off-Policy Learning: Importance Sampling
- 5 Off-policy Learning: Q-learning
- 6 Summary
- 7 References

# Outline

- 1 Introduction
- 2 On-Policy Monte-Carlo Control
- 3 On-Policy Temporal-Difference Learning
- 4 Off-Policy Learning: Importance Sampling
- 5 Off-policy Learning: Q-learning
- 6 Summary
- 7 References

# Model-Free Reinforcement Learning

- Last lecture:
  - ▶ Model-free prediction
  - ▶ Estimate the value function of an *unknown* MDP
- This lecture:
  - ▶ Model-free control
  - ▶ Optimize the value function of an *unknown* MDP

# Uses of Model-Free Control

Some example problems that can be modeled as MDPs

- Elevator
- Parallel Parking
- Ship Steering
- Bioreactor
- Helicopter
- Aeroplane Logistics
- Robocup Soccer
- Quake
- Portfolio management
- Protein Folding
- Robot walking
- Game of Go

For most of these problems, either:

- MDP model is unknown, but experience can be sampled
- MDP model is known, but is too big to use, except by samples

Model-free control can solve these problems

# On & Off-Policy Learning

$\neq$  Online / Offline

- Behavior-policy: determines which action to take, from which we determine the next state to visit (also called "sampling policy")
- Target-policy: determines the action that appears to be best (also called "learning policy")
- Goal in reinforcement learning:
  - ▶ improve the target policy
  - ▶ while using a behavior policy to ensure that we visit states often enough (exploration schemes such as  $\epsilon$ -greedy)
- On-policy learning: when the learning policy and the sampling policy are the same
- Off-policy learning: when the learning policy and the sampling policy are different

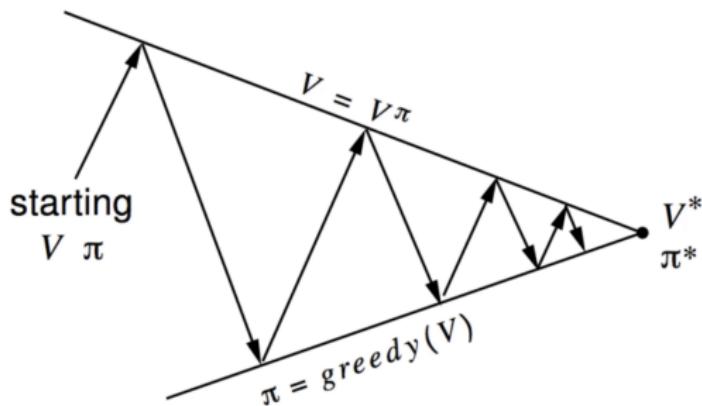
# On & Off-Policy Learning

- On-policy learning
  - ▶ "Learn on the job"
  - ▶ Learn the value of the target policy  $\pi$  from experience sampled from behavior policy  $\pi$
  - ▶ May not ensure enough exploration of state space
- Off-policy learning
  - ▶ "Look over someone's shoulder"
  - ▶ Learn the value of the target policy  $\pi$  from experience sampled from behavior policy  $\mu$
  - ▶ Learning is from experience(data) "off" the target policy
  - ▶ Compared to on-policy learning, off-policy learning is
    - ★ more powerful & general
    - ★ often of greater variance & slower to converge

# Outline

- 1 Introduction
- 2 On-Policy Monte-Carlo Control
- 3 On-Policy Temporal-Difference Learning
- 4 Off-Policy Learning: Importance Sampling
- 5 Off-policy Learning: Q-learning
- 6 Summary
- 7 References

# Generalized Policy Iteration (Refresher)

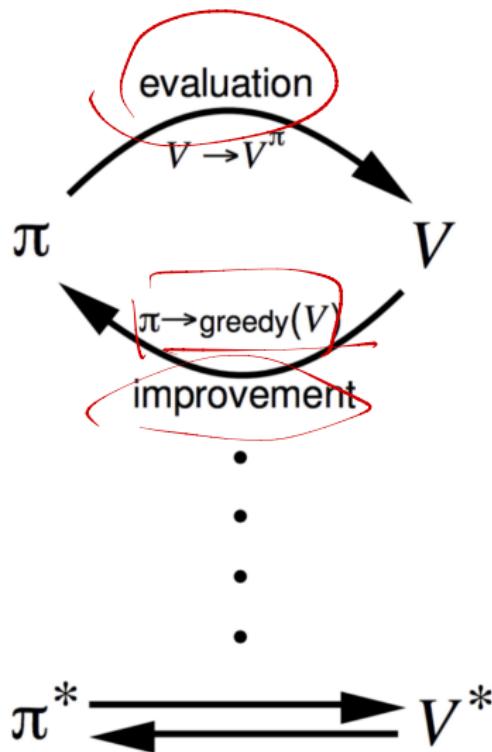


Policy evaluation Estimate  $v_\pi$

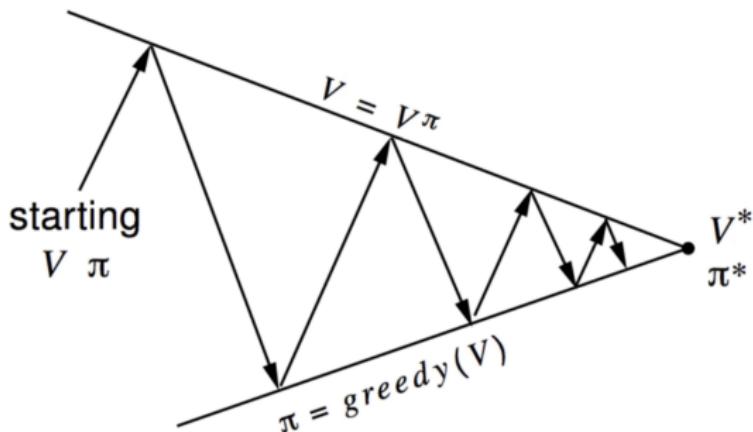
e.g. Iterative policy evaluation

Policy improvement Generate  $\pi' \geq \pi$

e.g. Greedy policy improvement



# Generalized Policy Iteration With Monte-Carlo Evaluation



Policy evaluation Monte-Carlo policy evaluation,  $V = v_\pi$ ?  
Policy improvement Greedy policy improvement?

# Model-Free Policy Iteration Using Action-Value Function

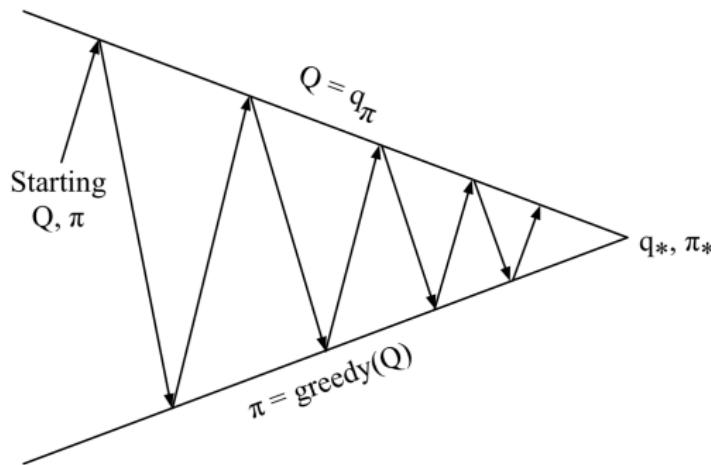
- Greedy policy improvement over  $V(s)$  requires model of MDP

$$\pi'(s) = \arg \max_{a \in \mathcal{A}} \{ \mathcal{R}_s^a + \mathcal{P}_{ss}^a V(s') \}$$

- Greedy policy improvement over  $Q(s, a)$  is model-free

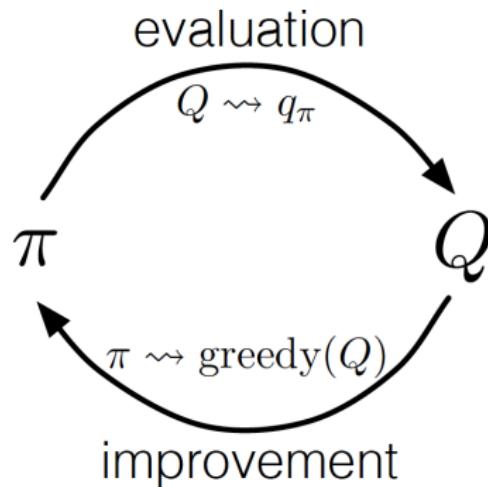
$$\pi'(s) = \arg \max_{a \in \mathcal{A}} Q(s, a)$$

# Generalized Policy Iteration with Action-Value Function



Policy evaluation Monte-Carlo policy evaluation,  $\underline{Q = q_\pi}$   
Policy improvement Greedy policy improvement?

# Monte-Carlo Control



- **MC policy iteration:** policy evaluation using MC methods followed by policy improvement
- **Policy improvement:** greedify with respect to value (or action-value) function

# Example of Greedy Action Selection



"Behind one door is tenure - behind the other is flipping burgers at McDonald's."

- There are two doors in front of you.
- You open the left door and get reward 0  
 $V(\text{left}) = 0$
- You open the right door and get reward +1  
 $V(\text{right}) = +1$
- You open the right door and get reward +3  
 $V(\text{right}) = +3$
- You open the right door and get reward +2  
 $V(\text{right}) = +2$
- Are you sure you've chosen the best door?

# $\epsilon$ -Greedy Exploration

- Tradeoff between exploitation and exploration
- All  $m$  actions are tried with non-zero probability
- With probability  $1 - \epsilon$  choose the greedy action
- With probability  $\epsilon$  choose an action at random

$$\pi(a|s) = \begin{cases} \frac{\epsilon}{m} + \frac{1-\epsilon}{\underset{a \in A}{\arg \max} Q(s, a)} & \text{if } a^* = \underset{a \in A}{\arg \max} Q(s, a) \\ \frac{\epsilon}{m} & \text{otherwise} \end{cases}$$

# $\epsilon$ -Greedy Policy Improvement

## Theorem

For any  $\epsilon$ -greedy policy  $\pi$ , the  $\epsilon$ -greedy policy  $\pi'$  with respect to  $q_\pi$  is an improvement,  $v_{\pi'}(s) \geq v_\pi(s)$

$$\begin{aligned}
 q_\pi(s, \pi'(s)) &= \sum_{a \in \mathcal{A}} \pi'(a|s) q_\pi(s, a) \\
 &= \frac{\epsilon}{m} \sum_{a \in \mathcal{A}} q_\pi(s, a) + (1 - \epsilon) \max_{a \in \mathcal{A}} q_\pi(s, a) \\
 &\geq \frac{\epsilon}{m} \sum_{a \in \mathcal{A}} q_\pi(s, a) + (1 - \epsilon) \underbrace{\sum_{a \in \mathcal{A}} \frac{\pi(a|s) - \epsilon/m}{1 - \epsilon} q_\pi(s, a)}_? \\
 &= \sum_{a \in \mathcal{A}} \pi(a|s) q_\pi(s, a) = v_\pi(s)
 \end{aligned}$$

Therefore from policy improvement theorem,  $v_{\pi'}(s) \geq v_\pi(s)$

## Remark

$$1^{\circ} \cdot \frac{\sum_{a \in A} \frac{r(s,a) - \varepsilon/m}{1-\varepsilon}}{1-\varepsilon} = \frac{\overbrace{\sum_{a \in A} r(s,a)} - \varepsilon}{1-\varepsilon} = \frac{1-\varepsilon}{1-\varepsilon} = 1$$

(A) = m

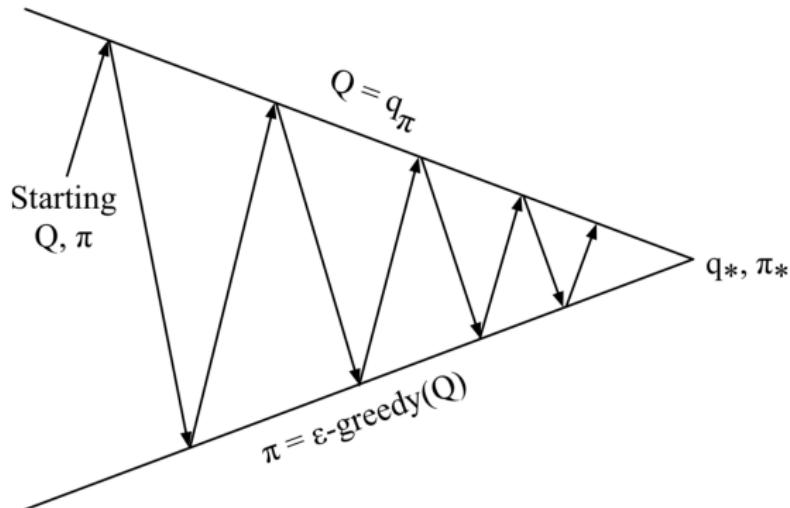
$$2^{\circ} \cdot \sum_{a \in A} \frac{r(s,a) - \varepsilon/m}{1-\varepsilon} \cdot q_{\pi}(s,a) \leq \sum_{a \in A} \frac{r(s,a) - \varepsilon/m}{1-\varepsilon} \cdot \max_{a \in A} q_{\pi}(s,a)$$

$\leftarrow$

$$= \max_{a \in A} q_{\pi}(s,a) \cdot \left( \frac{\sum_{a \in A} \frac{r(s,a) - \varepsilon/m}{1-\varepsilon}}{1} \right)$$

$$= \max_{a \in A} q_{\pi}(s,a)$$

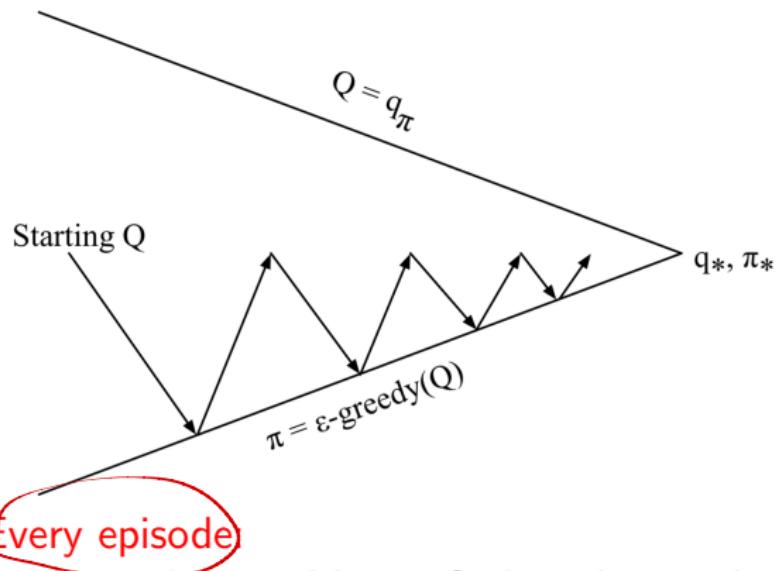
# Monte-Carlo Policy Iteration



Policy evaluation Monte-Carlo policy evaluation,  $Q = q_\pi$

Policy improvement  $\epsilon$ -greedy policy improvement

# Monte-Carlo Control



Policy evaluation Monte-Carlo policy evaluation,  $Q \approx q_\pi$

Policy improvement  $\epsilon$ -greedy policy improvement

## Definition

### Greedy in the Limit with Infinite Exploration (GLIE)

- All state-action pairs are explored infinitely many times,

$$\lim_{k \rightarrow \infty} N_k(s, a) = \infty$$

- The policy converges on a greedy policy,

$$\lim_{k \rightarrow \infty} \pi_k(a|s) = \mathbf{1}(a = \arg \max_{a' \in \mathcal{A}} Q_k(s, a'))$$

# GLIE

$$\frac{a}{bk+c}$$

- For example,  $\epsilon$ -greedy is GLIE if  $\epsilon$  reduces to zero at  $\epsilon_k = \frac{1}{k}$
- Keep a declining exploration probability at a sufficiently slow rate that try all actions infinitely often
- In practice,  $\epsilon_k = \frac{1}{k^\beta}$  with  $\beta \in (0.5, 1]$

$$\sum_{k=1}^{\infty} \epsilon_k = \infty$$
$$\sum_{k=1}^{\infty} \epsilon_k^2 < \infty$$

# GLIE Monte-Carlo Control

$$\mu = 2$$

- Sample  $k$ th episode using  $\pi$ :  $\{S_1, A_1, R_2, \dots, S_T\} \sim \underline{\pi}$
- For each state  $S_t$  and action  $A_t$  in the episode,

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

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

- Improve policy based on new action-value function

$$\epsilon \leftarrow 1/k$$

$$\pi \leftarrow \underline{\epsilon\text{-greedy}(Q)}$$

## Theorem

GLIE Monte-Carlo control converges to the optimal action-value function,  $Q(s, a) \rightarrow q_*(s, a)$

# On-Policy First-Visit MC Control

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

Algorithm parameter: small  $\varepsilon > 0$

Initialize:

$\pi \leftarrow$  an arbitrary  $\varepsilon$ -soft policy

$Q(s, a) \in \mathbb{R}$  (arbitrarily), for all  $s \in \mathcal{S}, a \in \mathcal{A}(s)$

$Returns(s, a) \leftarrow$  empty list, for all  $s \in \mathcal{S}, a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following  $\pi$ :  $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode,  $t = T-1, T-2, \dots, 0$ :

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair  $S_t, A_t$  appears in  $S_0, A_0, S_1, A_1, \dots, S_{t-1}, A_{t-1}$ :

Append  $G$  to  $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow$  average( $Returns(S_t, A_t)$ )

$A^* \leftarrow \arg \max_a Q(S_t, a)$  (with ties broken arbitrarily)

For all  $a \in \mathcal{A}(S_t)$ :

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

# Partial Summary of MC

- MC has several advantages over DP
  - ▶ can learn directly from interaction with environment
  - ▶ No need for full models
  - ▶ No need to learn about ALL states (no bootstrapping)
  - ▶ Less harmed by violating Markov property
- MC methods provide an alternate policy evaluation process
- One issue to watch for: maintaining sufficient exploration
  - ▶ exploring starts, soft policies

# Boltzmann Exploration

- A limitation of  $\epsilon$ -greedy: when we explore, we choose actions at random without regard to their estimated values
- For larger action spaces, this can mean that we are spending a lot of time evaluating actions that are quite poor
- Boltzmann Exploration (also known as “Gibbs sampling” & “soft-max”): choosing an action based on its estimated value.

$$\pi(a|s) \rightarrow \begin{cases} 1 & \text{if } a = a^* = \underset{a}{\operatorname{argmax}} \hat{Q}(s,a) \\ 0 & \text{otherwise} \end{cases}$$
$$\pi(a|s) = \frac{e^{\beta \hat{Q}(s,a)}}{\sum_{a'} e^{\beta \hat{Q}(s,a')}}$$

$\beta \rightarrow \infty$

$\pi(a|s) \propto -e^{-\beta \hat{Q}(s,a)}$

- $\hat{Q}(s, a)$  is an estimate of the value of being in state  $s$  and taking action  $a$ .
- $\beta$  is a tunable parameter
  - ▶  $\beta = 0$  produces a pure exploration policy
  - ▶  $\beta \rightarrow \infty$  produces a greedy policy

# Outline

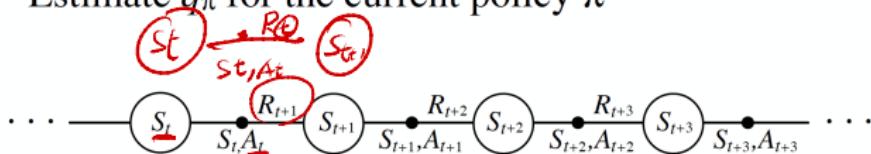
- 1 Introduction
- 2 On-Policy Monte-Carlo Control
- 3 On-Policy Temporal-Difference Learning
- 4 Off-Policy Learning: Importance Sampling
- 5 Off-policy Learning: Q-learning
- 6 Summary
- 7 References

# MC vs. TD Control

- Temporal-difference (TD) learning has several advantages over Monte-Carlo (MC)
  - ▶ Lower variance
  - ▶ Online
  - ▶ Incomplete sequences
- Natural idea: use TD instead of MC in our control loop
  - ▶ Apply TD to  $Q(S, A)$
  - ▶ Use  $\epsilon$ -greedy policy improvement
  - ▶ Update every time-step

# Learning An Action-Value Function

Estimate  $q_\pi$  for the current policy  $\pi$



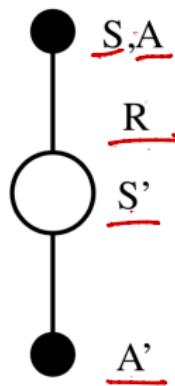
$$q_\pi(s, a) = E_\pi [R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t=s, A_t=a]$$

After every transition from a nonterminal state,  $S_t$ , do this:

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

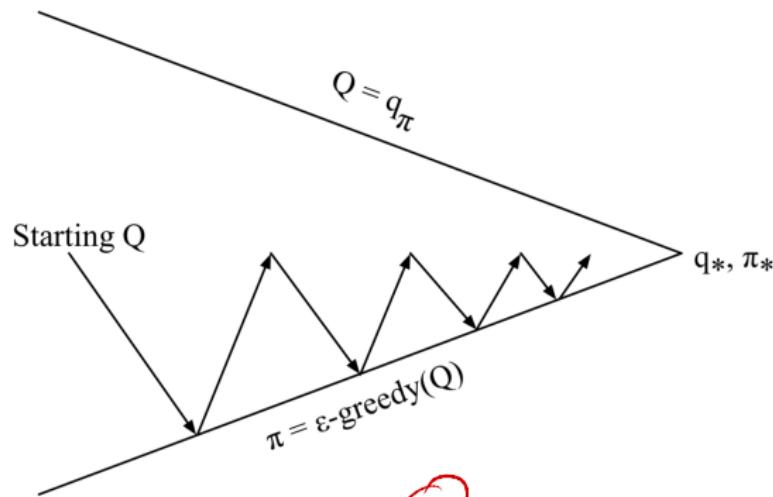
If  $S_{t+1}$  is terminal, then define  $Q(S_{t+1}, A_{t+1}) = 0$

# Updating Action-Value Functions with Sarsa



$$Q(S, A) \leftarrow Q(S, A) + \alpha(R + \gamma Q(S', A') - Q(S, A))$$

# On-Policy Control With Sarsa



Every time-step:

Policy evaluation Sarsa,  $Q \approx q_\pi$

Policy improvement  $\epsilon$ -greedy policy improvement

# Sarsa Algorithm for On-Policy Control

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  $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

    Initialize  $S$

    Choose  $A$  from  $S$  using policy derived from  $Q$  (e.g.,  $\varepsilon$ -greedy)

    Loop for each step of episode:

        Take action  $A$ , observe  $R, S'$

        Choose  $A'$  from  $S'$  using policy derived from  $Q$  (e.g.,  $\varepsilon$ -greedy)

$$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma Q(S', A') - Q(S, A)]$$

$S \leftarrow S'; A \leftarrow A'$ ;

    until  $S$  is terminal

# Convergence of Sarsa

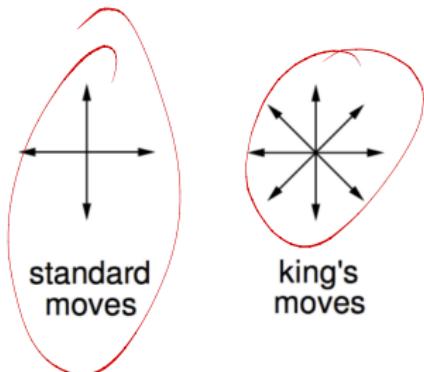
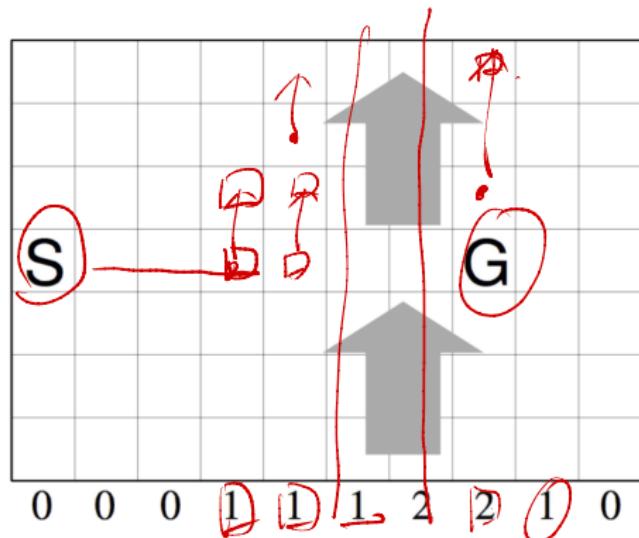
## Theorem

*Sarsa converges to the optimal action-value function,  $Q(s, a) \rightarrow q_*(s, a)$ , under the following conditions:*

- GLIE sequence of policies  $\pi_t(a|s)$
- Robbins-Monro sequence of step-sizes  $\alpha_t$

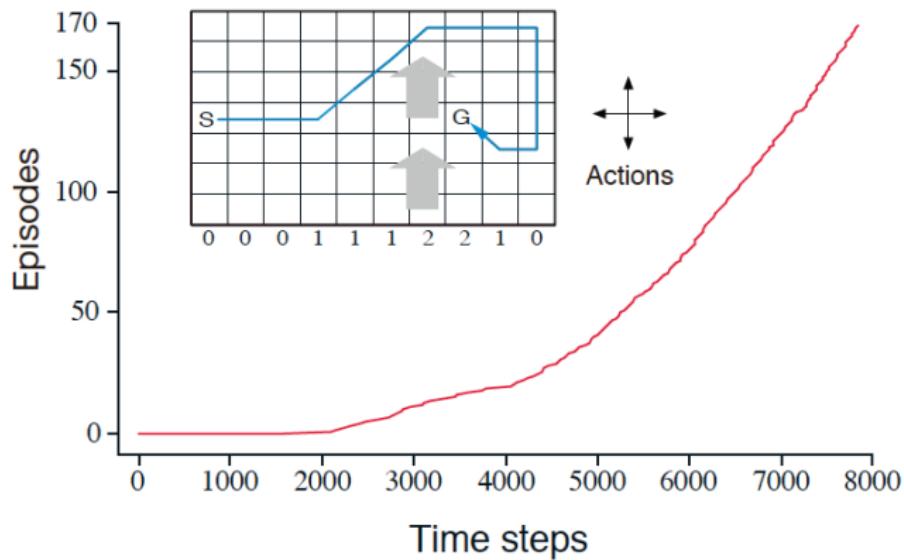
$$\sum_{t=1}^{\infty} \alpha_t = \infty$$
$$\sum_{t=1}^{\infty} \alpha_t^2 < \infty$$

# Windy Gridworld Example



- Reward = -1 per time-step until reaching goal
- Undiscounted

# Sarsa on the Windy Gridworld



## *n*-Step Sarsa

- Consider the following *n*-step returns for  $n = 1, 2, \infty$ :

$$n = 1 \quad (\text{Sarsa}) \quad q_t^{(1)} = R_{t+1} + \gamma Q(S_{t+1})$$

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

$$\vdots \qquad \vdots$$

$$n = \infty \quad (\text{MC}) \quad q_t^{(\infty)} = \underbrace{R_{t+1} + \gamma R_{t+2} + \dots + \gamma^{T-1} R_T}_{\text{...}}$$

- Define the *n*-step Q-return

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

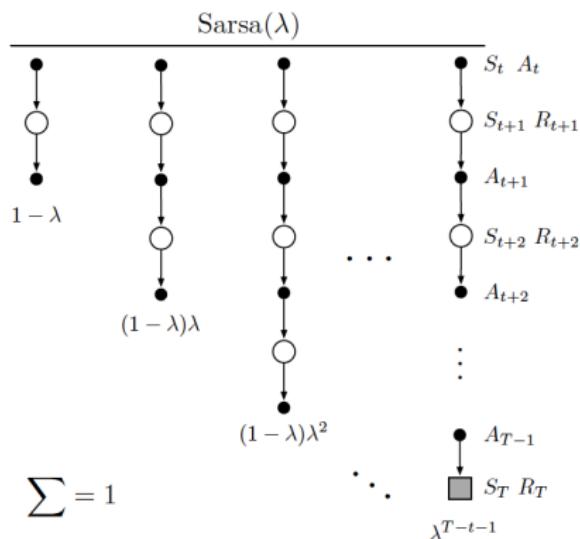
- n*-step Sarsa updates  $Q(s, a)$  towards the *n*-step Q-return

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left( q_t^{(n)} - Q(S_t, A_t) \right)$$

# Forward View Sarsa( $\lambda$ )

- Forward-view Sarsa( $\lambda$ )

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha (q_t^\lambda - Q(S_t, A_t))$$



- The  $q^\lambda$  return combines all n-step Q>Returns  $q_t^{(n)}$
- Using weight  $(1 - \lambda)\lambda^{n-1}$

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

# Outline

- 1 Introduction
- 2 On-Policy Monte-Carlo Control
- 3 On-Policy Temporal-Difference Learning
- 4 Off-Policy Learning: Importance Sampling
- 5 Off-policy Learning: Q-learning
- 6 Summary
- 7 References

# Off-Policy Learning

- Learn the value of the target policy  $\pi$  from experience due to behavior policy  $\mu$
- In this sense, learning is from experience(data) “off” the target policy, and the overall process is termed off-policy learning.
- For example,  $\pi$  is the greedy policy (and ultimately the optimal policy) while  $\mu$  is exploratory (e.g.,  $\epsilon$ -soft)
- In general, we only require coverage, i.e., that  $\mu$  generates behavior that covers, or includes,  $\pi$

$$\mu(a|s) > 0 \text{ for every } s, a \text{ at which } \pi(a|s) > 0$$

# Off-Policy Learning

- Evaluate target policy  $\pi(a|s)$  to compute  $v_\pi(s)$  or  $q_\pi(s, a)$
- While following behaviour policy  $\mu(a|s)$

$$\{S_1, A_1, R_2, \dots, S_T\} \sim \mu$$

- Why is this important?
  - ▶ Learn from observing humans or other agents
  - ▶ Re-use experience generated from old policies  $\pi_1, \pi_2, \dots, \pi_{t-1}$
  - ▶ Learn about optimal policy while following exploratory policy
  - ▶ Learn about multiple policies while following one policy

# Importance Sampling

- Estimate the expectation of a function

$$\mathbb{E}_{X \sim \pi}[g(X)] = \sum \pi(x) f(x) \approx \frac{1}{n} \sum_{k=1}^n g(x_k), x_k \sim \pi$$

- But sometimes it is difficult to sample  $x$  from  $\pi$ , then we can sample  $x$  from another distribution  $\mu$ , then correct the weight

$$\begin{aligned}\mathbb{E}_{X \sim \pi}[g(X)] &= \sum \pi(x) g(x) = \sum \mu(x) \frac{\pi(x)}{\mu(x)} g(x) \\ &= \mathbb{E}_{X \sim \mu} \left[ \frac{\pi(X)}{\mu(X)} g(X) \right] \approx \frac{1}{n} \sum_{k=1}^n \left( \frac{\pi(x_k)}{\mu(x_k)} g(x_k) \right), x_k \sim \mu\end{aligned}$$

- For off-policy learning: weight each return by the ratio of the probabilities of the trajectory under the two policies

# Importance Sampling for Off-Policy Monte-Carlo

- Use returns generated from  $\mu$  to evaluate  $\pi$
- Weight return  $G_t$  according to similarity between policies
- Multiply importance sampling corrections along whole episode

$$G_t^{\pi/\mu} = \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)} \cdot \frac{\pi(A_{t+1}|S_{t+1})}{\mu(A_{t+1}|S_{t+1})} \cdots \frac{\pi(A_T|S_T)}{\mu(A_T|S_T)} G_t$$

- Update value towards corrected return

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

- Cannot use if  $\mu$  is zero when  $\pi$  is non-zero
- Importance sampling can dramatically increase variance

# Importance Sampling Ratio

- Probability of the rest of the trajectory, after  $S_t$ , under  $\pi$

$$\begin{aligned} & \Pr \{ A_t, S_{t+1}, A_{t+1}, \dots, S_T | S_t, A_{t:T-1} \sim \pi \} \\ &= \underbrace{\pi(A_t|S_t)}_{\substack{\uparrow \text{Policy}}} p(S_{t+1}|S_t, A_t) \underbrace{\pi(A_{t+1}|S_{t+1})}_{\substack{\uparrow \text{Model info.}}} \cdots p(S_T|S_{T-1}, A_{T-1}) \\ &= \prod_{k=t}^{T-1} \underbrace{\pi(A_k|S_k)}_{\substack{\uparrow \text{Policy}}} p(S_{k+1}|S_k, A_k), \end{aligned}$$

- In importance sampling, each arm is weighted by the relative probability of the trajectory under the two policies

$$\rho_t^T = \frac{\prod_{k=t}^{T-1} \pi(A_k|S_k) p(S_{k+1}|S_k, A_k)}{\prod_{k=t}^{T-1} \mu(A_k|S_k) p(S_{k+1}|S_k, A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{\mu(A_k|S_k)}$$

Model-free

- This is called the importance sampling ratio
- All importance sampling ratios have expected value 1

$$\mathbb{E}_{A_k \sim \mu} \left[ \frac{\pi(A_k|S_k)}{\mu(A_k|S_k)} \right] = \sum_a \mu(a|S_k) \frac{\pi(a|S_k)}{\mu(a|S_k)} = \sum_a \pi(a|S_k) = 1.$$

# Off-Policy MC Policy Evaluation

Incremental off-policy every-visit MC policy evaluation (returns  $Q \approx q_\pi$ )

Input: an arbitrary target policy  $\pi$

Initialize, for all  $s \in \mathcal{S}$ ,  $a \in \mathcal{A}(s)$ :

$$Q(s, a) \leftarrow \text{arbitrary}$$

$$C(s, a) \leftarrow 0$$

Repeat forever:

$\mu \leftarrow$  any policy with coverage of  $\pi$

Generate an episode using  $\mu$ :

$$S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T, S_T$$

$$G \leftarrow 0$$

$$W \leftarrow 1$$

For  $t = T - 1, T - 2, \dots$  downto 0:

$$G \leftarrow \gamma G + R_{t+1}$$

$$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$$

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

$$W \leftarrow W \frac{\pi(A_t | S_t)}{\mu(A_t | S_t)}$$

If  $W = 0$  then ExitForLoop

# Off-Policy MC Control

Off-policy every-visit MC control (returns  $\pi \approx \pi_*$ )

Initialize, for all  $s \in \mathcal{S}$ ,  $a \in \mathcal{A}(s)$ :

$$Q(s, a) \leftarrow \text{arbitrary}$$

$$C(s, a) \leftarrow 0$$

$$\pi(s) \leftarrow \arg \max_a Q(S_t, a) \quad (\text{with ties broken consistently})$$

Repeat forever:

$$\mu \leftarrow \text{any soft policy}$$

Generate an episode using  $\mu$ :

$$S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T, S_T$$

$$G \leftarrow 0$$

$$W \leftarrow 1$$

For  $t = T - 1, T - 2, \dots$  downto 0:

$$G \leftarrow \gamma G + R_{t+1}$$

$$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$$

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

$$\pi(S_t) \leftarrow \arg \max_a Q(S_t, a) \quad (\text{with ties broken consistently})$$

If  $A_t \neq \pi(S_t)$  then ExitForLoop

$$W \leftarrow W u(A_t | S_t)$$

Target policy is greedy and deterministic

Behavior policy is soft, typically  $\epsilon$ -greedy

# Importance Sampling for Off-Policy TD

n-Step TD

TD( $\pi$ )

- Use TD targets generated from  $\mu$  to evaluate  $\pi$
- Weight TD target  $R + \gamma V(S')$  by importance sampling
- Only need a single importance sampling correction

$$V(S_t) \leftarrow V(S_t) + \alpha \left( \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)} (R_{t+1} + \gamma V(S_{t+1})) - V(S_t) \right)$$

- Much lower variance than Monte-Carlo importance sampling
- Policies only need to be similar over a single step

# Outline

- 1 Introduction
- 2 On-Policy Monte-Carlo Control
- 3 On-Policy Temporal-Difference Learning
- 4 Off-Policy Learning: Importance Sampling
- 5 Off-policy Learning: Q-learning
- 6 Summary
- 7 References

# Q-Learning

- We now consider off-policy learning of action-values  $Q(s, a)$
- **No** importance sampling is required
- Next action to evaluate is chosen using behavior policy  
 $A_{t+1} \sim \mu(\cdot | S_t)$
- But we consider alternative successor action  $A' \sim \pi(\cdot | S_t)$
- Which means a separate policy is used to choose the alternative action in the future
- And update  $Q(S_t, A_t)$  towards value of alternative action

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma Q(S_{t+1}, \underline{A'}) - Q(S_t, A_t))$$

$\underline{A_{t+1}}$

# Off-Policy Control with Q-Learning

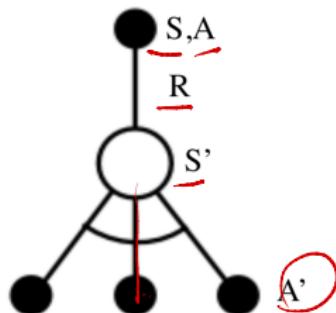
- We now allow both behaviour and target policies to **improve**
- The target policy  $\pi$  is **greedy** w.r.t.  $\underline{Q(s, a)}$

$$\pi(S_{t+1}) = \arg \max_{a'} Q(S_{t+1}, a')$$

- The behaviour policy  $\mu$  is e.g.  **$\epsilon$ -greedy** w.r.t.  $\underline{Q(s, a)}$
- The Q-learning target then simplifies:

$$\begin{aligned} & R_{t+1} + \gamma \underline{Q(S_{t+1}, A')} \\ &= R_{t+1} + \gamma Q(S_{t+1}, \arg \max_{a'} Q(S_{t+1}, a')) \\ &= R_{t+1} + \max_{a'} \gamma \underline{Q(S_{t+1}, a')} \end{aligned}$$

# Q-Learning Control Algorithm



$$Q(S, A) \leftarrow Q(S, A) + \alpha \left( R + \gamma \max_{a'} Q(S', a') - Q(S, A) \right)$$

## Theorem

*Q-learning control converges to the optimal action-value function,  
 $Q(s, a) \rightarrow q_*(s, a)$*

# Q-Learning Algorithm for Off-Policy Control

Why ~~no~~ Importance sampling ?

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  $Q(\text{terminal}, \cdot) = 0$

Loop for each episode:

    Initialize  $S$

    Loop for each step of episode:

        Choose  $A$  from  $S$  using policy derived from  $Q$  (e.g.,  $\varepsilon$ -greedy)

        Take action  $A$ , observe  $R, S'$

$$Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \max_a Q(S', a) - Q(S, A)]$$

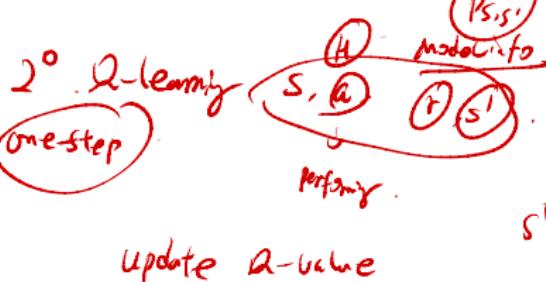
$S \leftarrow S'$

    until  $S$  is terminal

# Q-Learning Demo

1°. Q-learning Value-iteration

E<sub>1</sub>(t)



Sampling from  $\mu$  to compute  $E_2$

<https://www.cs.ubc.ca/~poole/demos/rl/q.html>

3°. one-step Q-learning  $Q^*(s, a) = E \left[ \underset{s' \sim P_{s,a}}{R_{s'}} + \gamma \max_{a'} Q^*(s', a') \right]$

Importance sampling  
 $| s_t=s, A_t=a$

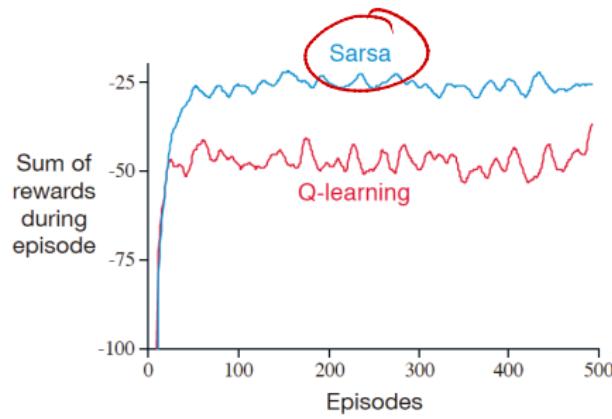
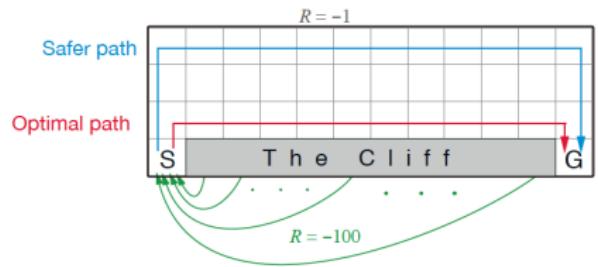
4°. two-step Q-learning  $Q^*(s, a)$

$$= E \left[ R_{s'} + \gamma R_{s''} + \gamma^2 \max_{a''} Q^*(s'', a'') \right] | \begin{array}{l} s_t=s, \\ A_t=a \end{array}$$

$s' \sim P_{s,a}, s'' \sim P_{s',a'}$

Need importance sampling ratio

# Cliff Walking Example



# Outline

- 1 Introduction
- 2 On-Policy Monte-Carlo Control
- 3 On-Policy Temporal-Difference Learning
- 4 Off-Policy Learning: Importance Sampling
- 5 Off-policy Learning: Q-learning
- 6 Summary
- 7 References

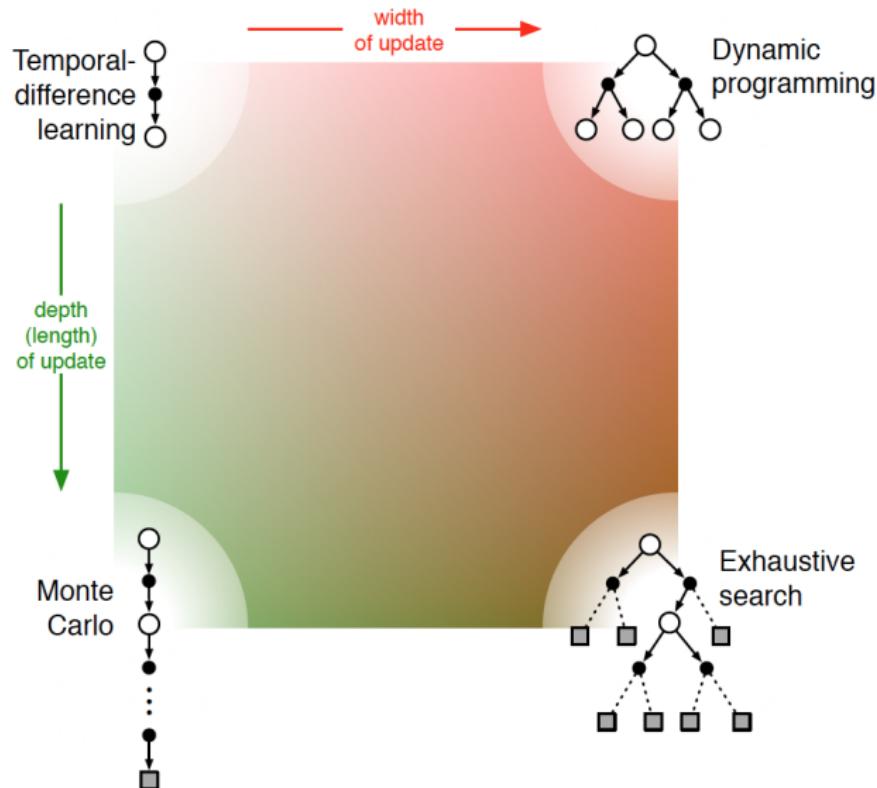
# Relationship Between DP and TD

	<u>Full Backup (DP)</u>	<u>Sample Backup (TD)</u>
Bellman Expectation Equation for $v_\pi(s)$	<p><math>v_\pi(s) \leftarrow s</math></p> <p><math>v_\pi(s') \leftarrow s'</math></p> <p>Iterative Policy Evaluation</p>	<p>TD Learning</p>
Bellman Expectation Equation for $q_\pi(s, a)$	<p><math>q_\pi(s, a) \leftarrow s, a</math></p> <p><math>q_\pi(s', a') \leftarrow a'</math></p> <p>Q-Policy Iteration</p>	<p>Sarsa</p>
Bellman Optimality Equation for $q_*(s, a)$	<p><math>q_*(s, a) \leftarrow s, a</math></p> <p><math>q_*(s', a') \leftarrow a'</math></p> <p>Q-Value Iteration</p>	<p>Q-Learning</p>

# Relationship Between DP and TD

<i>Full Backup (DP)</i>	<i>Sample Backup (TD)</i>
Iterative Policy Evaluation $V(s) \leftarrow \mathbb{E}[R + \gamma V(S')   s]$	TD Learning $V(S) \xleftarrow{\alpha} R + \gamma V(S')$
Q-Policy Iteration $Q(s, a) \leftarrow \mathbb{E}[R + \gamma Q(S', A')   s, a]$	Sarsa $Q(S, A) \xleftarrow{\alpha} R + \gamma Q(S', A')$
<u>Q-Value Iteration</u> $Q(s, a) \leftarrow \mathbb{E} \left[ R + \gamma \max_{a' \in \mathcal{A}} Q(S', a')   s, a \right]$	Q-Learning $Q(S, A) \xleftarrow{\alpha} R + \gamma \max_{a' \in \mathcal{A}} Q(S', a')$

# Unified View of Reinforcement Learning



# Outline

- 1 Introduction
- 2 On-Policy Monte-Carlo Control
- 3 On-Policy Temporal-Difference Learning
- 4 Off-Policy Learning: Importance Sampling
- 5 Off-policy Learning: Q-learning
- 6 Summary
- 7 References

# Main References

- Reinforcement Learning: An Introduction (second edition), R. Sutton & A. Barto, 2018.
- RL course slides from Richard Sutton, University of Alberta.
- RL course slides from David Silver, University College London.