Model-Free Control

Junni Zou

Institute of Media, Information and Network
Dept. of Computer Science and Engineering
Shanghai Jiao Tong University
http://min.sjtu.edu.cn

Spring, 2021

Institute of Media, Information, and Network

Outline

Table of Contents

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

Institute of Media, Information, and Network

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

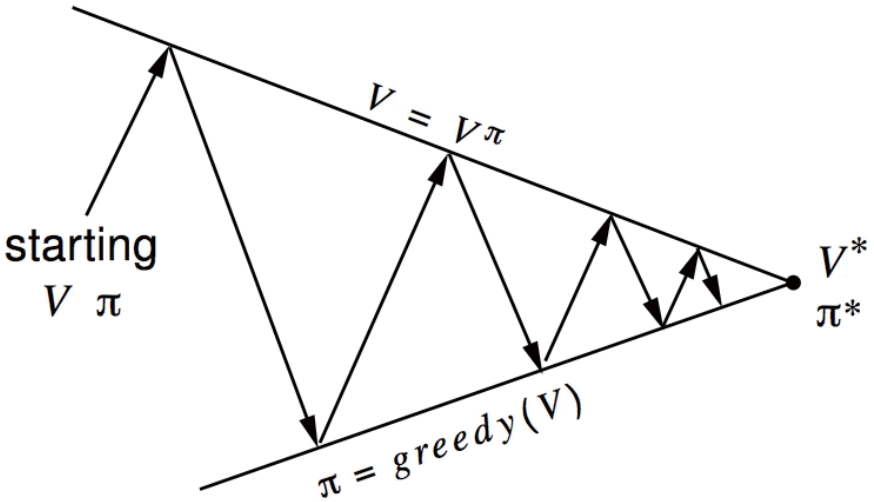
Model-free control can solve these problems

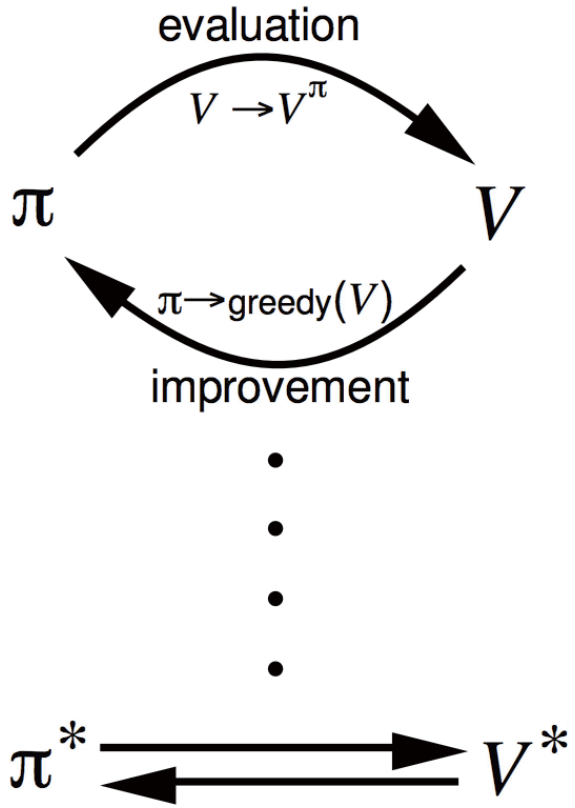
On and Off-Policy Learning

- On-policy learning
  - "Learn on the job"
  - Learn about policy π from experience sampled from π
- Off-policy learning
  - "Look over someone's shoulder"
  - Learn about policy π from experience sampled from μ

Table of Contents

Generalized Policy Iteration (Refresher)

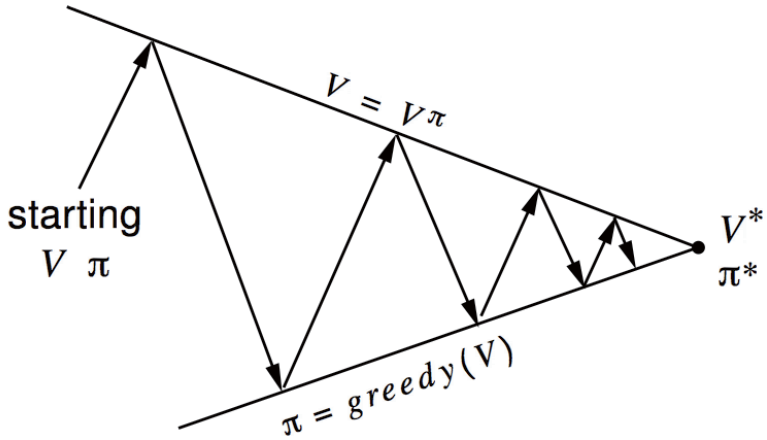




Policy evaluation Estimate v_π
e.g. Iterative policy evaluation
Policy improvement Generate $\pi' \geq \pi$
e.g. Greedy policy improvement

Institute of Media, Information, and Network

Generalized Policy Iteration With Monte-Carlo Evaluation



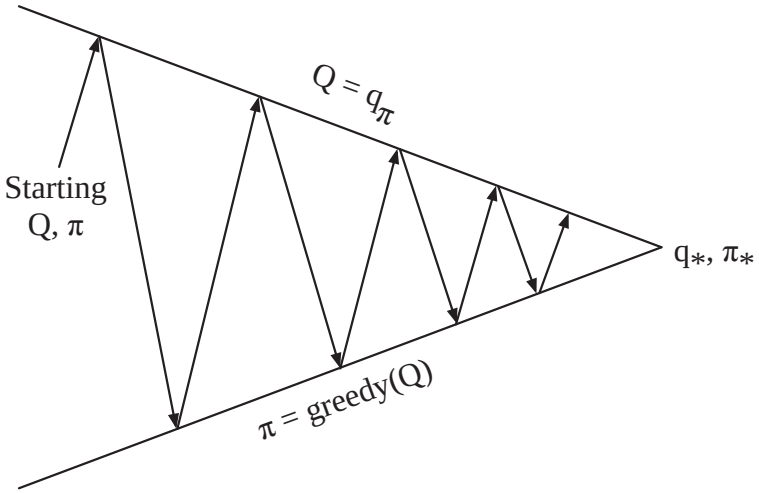
Greedy policy improvement over $V(s)$ requires model of MDP

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

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

Institute of Media, Information, and Network

Generalized Policy Iteration with Action-Value Function



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

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

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

Policy improvement Greedy policy improvement?

Institute of Media, Information, and Network

Example of Greedy Action Selection

- There are two doors in front of you
- You open the left door and get reward 0
  $V(left) = 0$
- You open the right door and get reward +1
  $V(right) = +1$
- You open the right door and get reward +3
  $V(right) = +2\ ((1+3)/2)$
- You open the right door and get reward +2
  $V(right) = +2\ ((1+3+2)/3)$

...

- Are you sure you've chosen the best door?

Institute of Media, Information, and Network

ϵ-Greedy Exploration

- Simplest idea for ensuring continual exploration
- All m actions are tried with non-zero probability
- With probability $1-\epsilon$ choose the greedy action
- With probability ϵ choose an action at random

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

Institute of Media, Information, and Network

ϵ-Greedy Improvement

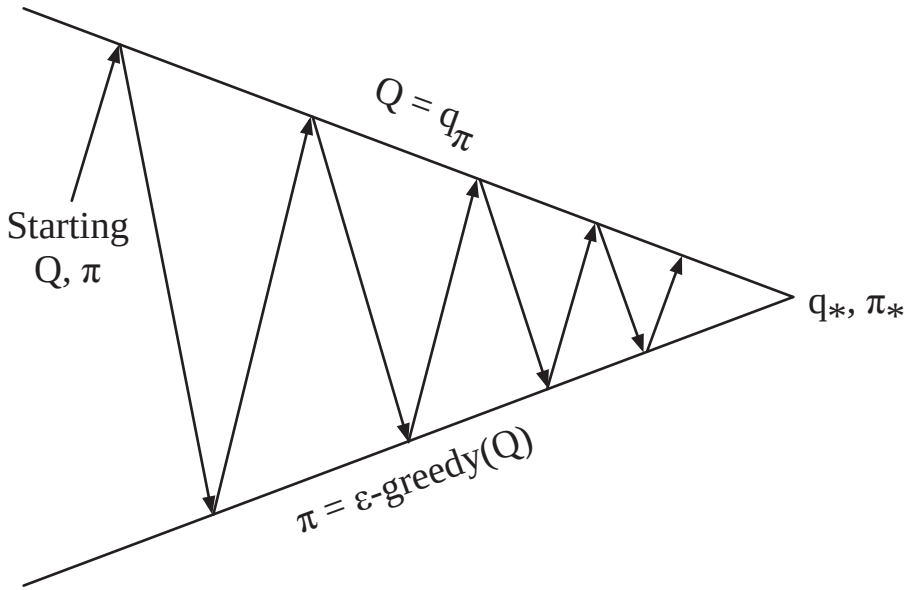
Theorem

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

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

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

Monte-Carlo Policy Iteration



Every episode:

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

Policy improvement ϵ-greedy policy improvement

Institute of Media, Information, and Network

Monte-Carlo Policy Iteration (2)

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:
$G \leftarrow$ return following the first occurrence of s, a
Append G to $Returns(s,a)$
$Q(s,a) \leftarrow \text{average}(Returns(s,a))$
(c) For each s in the episode:
$A^* \leftarrow \arg\max_a Q(s,a)$
For all $a \in \mathcal{A}(s)$:

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

Information, and Network

GLIE

Definition

Greedy in the Limit with Infinite Exploration (GLIE)

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

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

- The policy converges on a greedy policy,

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

For example, ϵ-greedy is GLIE if ϵ reduces to zero at $\epsilon_k = \frac{1}{k}$

Institute of Media, Information, and Network

GLIE Monte-Carlo Control

- Sample k-th episode using $\pi : \{S_1, A_1, R_2, \cdots, S_T\} \sim \pi$
- 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

$$\epsilon \leftarrow 1/k$$

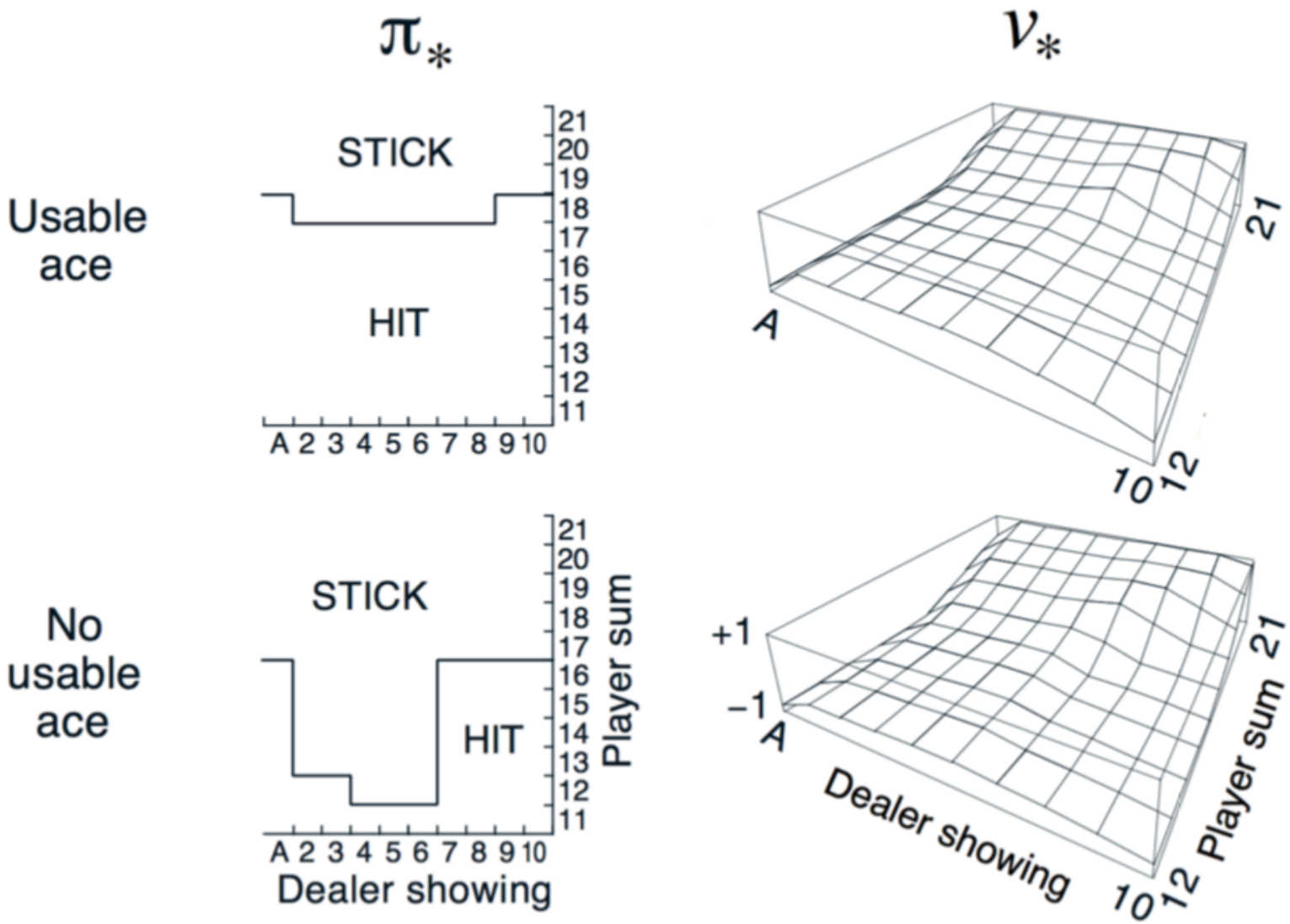
$$\pi \leftarrow \epsilon - \text{greedy}(Q)$$

Theorem

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

Back to the Blackjack Example

Monte-Carlo Control in Blackjack



Media,
n, and Network

Monte-Carlo Control in Blackjack (2)

最优策略是这样：当你手上有可用 A 时，大多数情况下当你的牌面和达到17或18时停止要牌，如果庄家可见的牌面在2-9之间，你选择17，其它条件选择18；当你手上没有 A 时，最优策略提示大多数情况下牌面和达到16就要停止叫牌，当庄家可见的牌面在2-7时，这一数字更小至13甚至12。这种极端情况下，宁愿停止叫牌等待让庄家的牌爆掉。

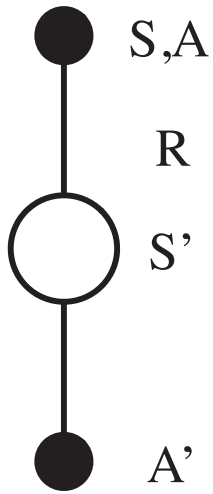
本文对于使用该策略进行赌博导致的输赢不负任何责任。

Table of Contents

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 ϵ-greedy policy improvement
  - Update every time-step

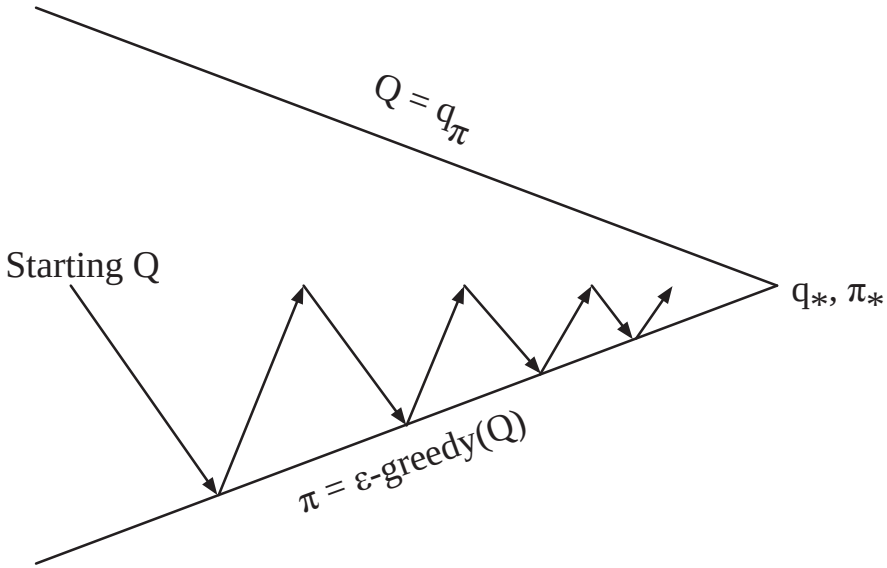
Updating Action-Value Functions with Sarsa



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

Institute of Media, Information, and Network

On-Policy Control With Sarsa



Every time-step:

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

Policy improvement ϵ-greedy policy improvement

Institute of Media, Information, and Network

Sarsa Algorithm for On-Policy Control

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)
    $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma Q(S',A') - Q(S,A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

Convergence of Sarsa

Theorem

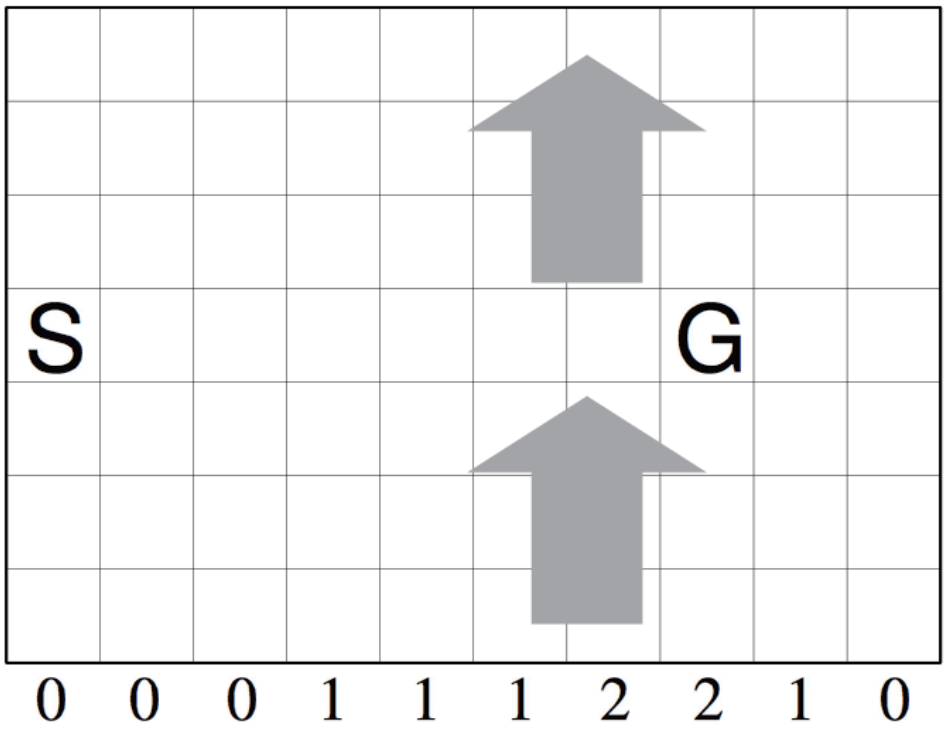
Sarsa converges to the optimal action-value function, $Q(s,a) \to q_*(s,a)$, *under the following conditions:*

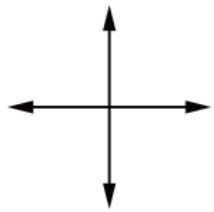
- GLIE sequence of policies $\pi_t(a|s)$
- Robbins-Monro sequence of step-sizes α_t

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

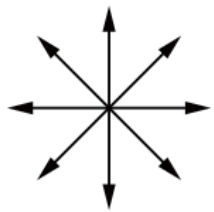
$$\sum_{t=1}^{\infty} \alpha_t^2 < \infty$$

Windy Gridworld Example





standard moves



king's moves

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

Institute of Media, Information, and Network

Windy Gridworld Example (2)

已知：如图所示，环境是一个10*7的长方形格子世界，同时有一个起始位置S和一个终止目标位置G，水平下方的数字表示对应的列中有一定强度的风，当该数字是1时，个体进入该列的某个格子时，会按图中箭头所示的方向自动移动一格，当数字为2时，表示顺风移动2格，以此类推模拟风的作用。任何试图离开格子世界的行为都会使得个体停留在移动前的位置。对于个体来说，它不清楚整个格子世界的构造，即它不知道格子是长方形的，也不知道边界在哪里。也不清楚起始位置、终止目标位置的具体为止。对于它来说，每一个格子就相当于一个封闭的房间，在没推开门离开当前房间之前它无法知道会进入哪个房间。个体具备记住曾经去过的格子的能力。格子可以执行的行为是朝上、下、左、右移动一步。

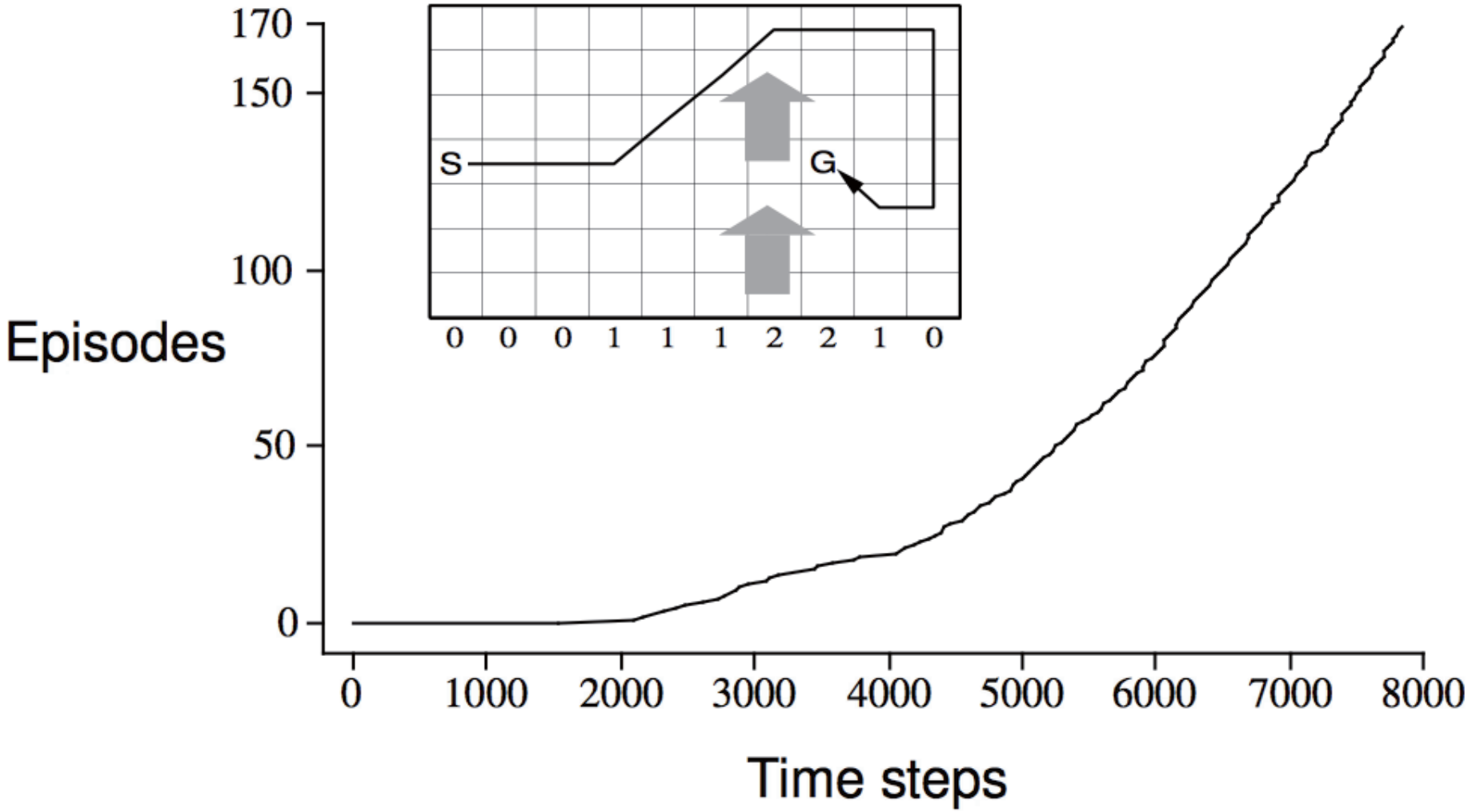
问题：个体如何才能找到最短从起始格子S到终止目标格子G的最短路线?

Institute of Media, Information, and Network

Windy Gridworld Example (3)

解答： 首先将这个问题用强化学习常用的语言重新描述下。这是一个不基于模型的控制问题，即个体在不清楚模型机制条件下试图寻找最优策略的问题。在这个问题中，环境信息包括格子世界的形状是10*7的长方形；起始和终止格子的位置，可以用二维或一维的坐标描述，同时还包括个体在任何时候所在的格子位置。风的设置是环境动力学的一部分，它与长方形的边界共同及个体的行为共同决定了个体下一步的状态。个体从环境观测不到自身位置、起始位置以及终止位置信息的坐标描述，个体在与环境进行交互的过程中学习到自身及其它格子的位置关系。个体的行为空间是离散的四个方向。可以设置个体每行走一步获得即时奖励为-1，直到到达终止目标位置的即时奖励为0，借此希望找到最优策略。衰减系数λ可设为1。

Sarsa on the Windy Gridworld



n-Step Sarsa

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

$$
\begin{array}{lll}
n = 1 & \textit{Sarsa} & q_t^{(1)} = R_{t+1} + \gamma Q(S_{t+1}) \\
n = 2 & & q_t^{(2)} = R_{t+1} + \gamma R_{t+2} + \gamma^2 Q(S_{t+2}) \\
\vdots & & \vdots \\
n = \infty & (MC) & q_t^{\infty} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_T
\end{array}
$$

- Define the n-step Q-return

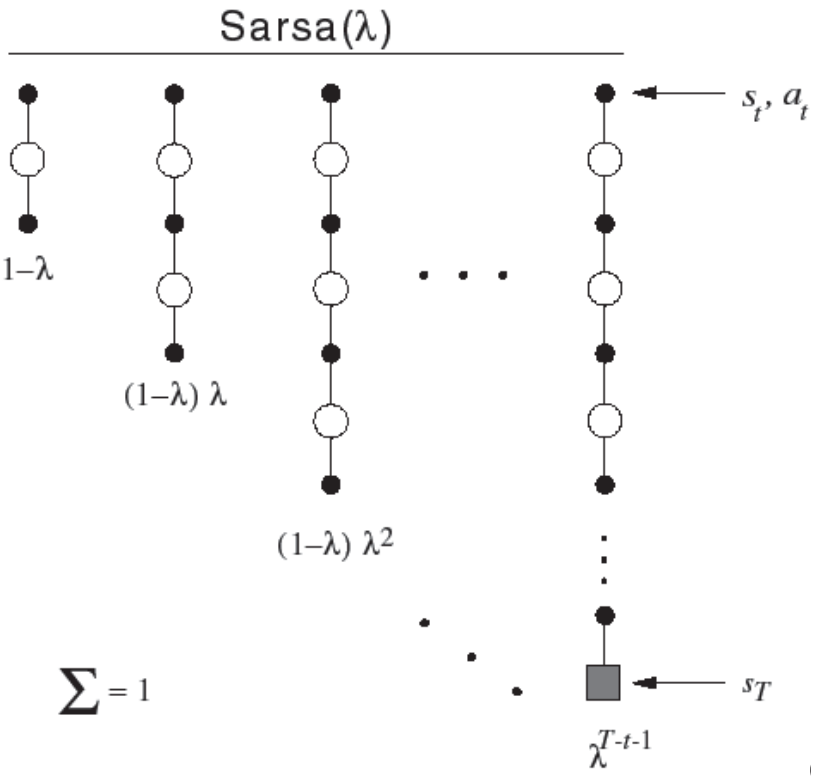
$$q_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n Q(S_{t+n})$$

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

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

Institute of Media, Information, and Network

Forward View Sarsa(λ)



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

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

- Forward-view Sarsa(λ)

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

Institute of Media, Information, and Network

Backward View Sarsa(λ)

- Just like $TD(\lambda)$, we use eligibility traces in an online algorithm
- But Sarsa(λ) has one eligibility trace for each state-action pair

$$E_0(s,a) = 0$$
$$E_t(s,a) = \gamma\lambda E_{t-1}(s,a) + \mathbf{1}(S_t = s, A_t = a)$$

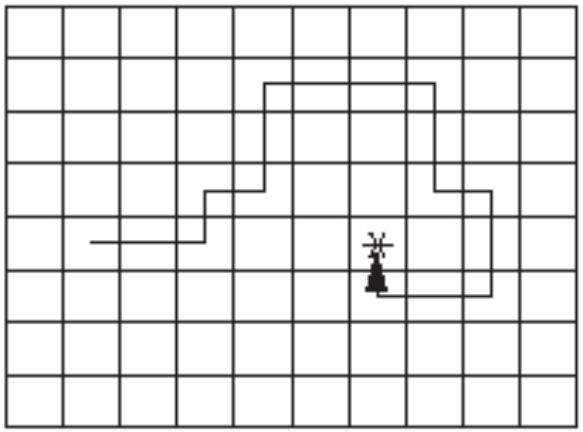
- $Q(s,a)$ is updated for every state s and action a
- In proportion to TD-error δ_t and eligibility trace $E_t(s,a)$

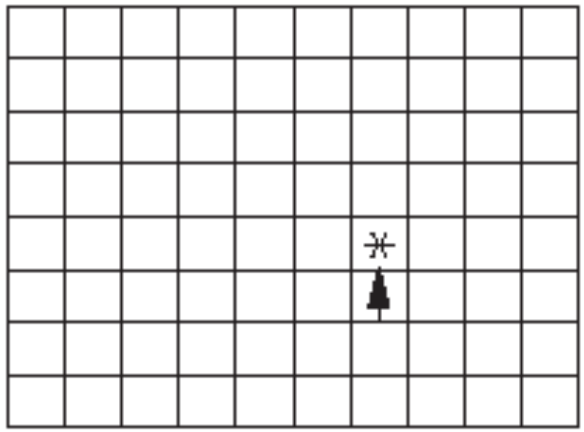
$$\delta_t = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)$$
$$Qs, a \leftarrow Q(s,a) + \alpha\delta_t E_t(s,a)$$

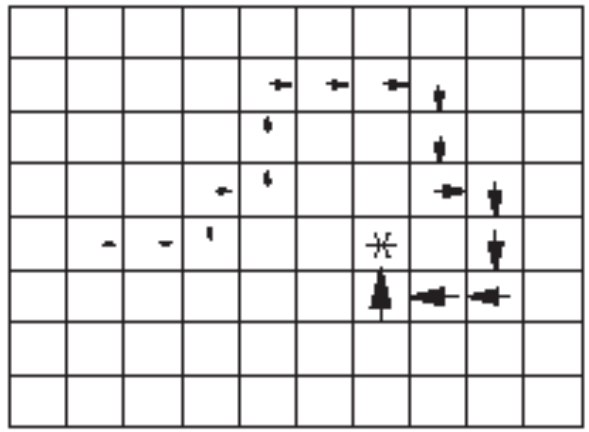
Sarsa(λ) Algorithm

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 $s \in \mathcal{S}, a \in \mathcal{A}(s)$
  Initialize S, 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 $s \in \mathcal{S}, a \in \mathcal{A}(s)$:
      $Q(s,a) \leftarrow Q(s,a) + \alpha \delta E(s,a)$
      $E(s,a) \leftarrow \gamma \lambda E(s,a)$
    $S \leftarrow S'$; $A \leftarrow A'$
  until S is terminal

Sarsa Gridworld Example







先做几个合理的小约定：1）认定每一步的即时奖励为0，直到终点处即时奖励为1；2）根据算法，除了终点以外的任何状态行为对的Q值可以是任意的，但我们设定所有的Q值均为0；3）该路线是第一次找到终点的路线。

Sarsa Gridworld Example (2)

由于是现时策略学习，一开始个体对环境一无所知，即 Q 值均为0，它将随机选取移步行为。在到达终点前的每一个位置 S，个体依据当前策略，产生一个移步行为，执行该行为，环境会将其放置到一个新位置 S'，同时给以即时奖励0，在新的位置 S' 上，根据当前的策略它会产生新位置下的一个行为，个体不执行该行为，仅仅在表中查找新状态下新行为的 Q' 值，由于 $Q=0$，依据更新公式，它将把刚才离开的位置以及对应的行为的状态行为对价值 $Q<S,A>$ 更新为0。如此直到个体最到达终点位置 S_G，它获得一个即时奖励1，此时个体会依据公式更新其到达终点位置前所在那个位置（暂用 S_H 表示，也就是图中终点位置下方，向上的箭头所在的位置）时采取向上移步的那个状态行为对价值 $Q<S_H,A_{up}>$ 值，它将不再是0，这是个体在这个Episode中唯一一次用非0数值来更新Q值。这样完成一个Episode，此时个体已经并只进行了**一次有意义的行为**价值函数的更新；同时依据新的价值函数产生了新的策略。这个策略绝大多数与之前的相同，只是当个体处在某一个特殊位置时将会有一个确定的行为：直接向上移步，这个位置就是与终点相邻的下方的格子。这里请不要误认为Sarsa算法只在经历一个完整的Episode之后才更新，在这个例子中，由于我们的设定，它每走一步都会更新，只是多数时候更新的数据和原来一样罢了。

Sarsa Gridworld Example (3)

此时如果要求个体继续学习，则环境将其放入起点。个体的第二次寻路过程一开始与首次一样都是盲目随机的，直到其进入终点位置下方的位置 S_H，在这个位置，个体更新的策略要求其选择向上的行为直接进入终点位置 S_G。

同样，经过第二次的寻路，个体了解到到达终点下方的位置 S_H 价值比较大，因为在这个位置直接采取向上移步的行为就可以拿到到达终点的即时奖励。因此它会将其通过移动一步可以到达 S_H 的其它位置以及相应的到达 S_H 位置索要采取的行为这一状态行为对的价值提升。如此反复，[illegible]采用greedy策略更新，个体最终将得到一条到达终点的路径，不过这条路径的倒数第二步永远在终点位置的下方。如果采用ε-greedy策略更新，那么个体还会尝试到终点位置的左上右等其它方向的相邻位置价值也比较大，此时个体每次完成的路径可能都不一样。通过重复多次搜索，这种Q值的实质性的更新将覆盖越来越多的状态行为对，个体在早期采取的随机行为的步数将越来越少，直至最终实质性的更新覆盖到起始位置。此时个体将能直接给出一条确定的从起点到终点的路径。

Table of Contents

Off-Policy Learning

- Learning control methods face a dilemma
  - Learn action values on subsequent *optimal* behavior
  - Behave *non-optimally* to explore all actions
- How can learn about optimal policy while following exploratory policy?
- On-policy learning: use ϵ-greedy improvement
- Off-policy learning: use two policies $\pi(a|s)$ and $\mu(a|s)$
  - Evaluate target policy $\pi(a|s)$ to compute $q_\pi(s, a)$
  - While following behavior policy $\mu(a|s)$

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

  - Learn from observing humans or other agents
  - Learning is from data "off" the target policy

Institute of Media, Information, and Network

Monte-Carlo Policy Iteration (2)

On-policy v.s. Off-policy

- On-policy: The agent learned and the agent interacting with the environment is the same.
- Off-policy: The agent learned and the agent interacting with the environment is different.

Monte-Carlo Policy Iteration (2)

On-policy v.s. Off-policy

- On-policy: The agent learned and the agent interacting with the environment is the same.
- Off-policy: The agent learned and the agent interacting with the environment is different.

阿光下棋

佐為下棋、阿光在旁邊看

edia,
nd Network

On-Policy vs. Off-Policy

- On-policy is generally simple
- Off-policy is often of greater variance and slower convergence
- Off-policy is more powerful and general
- On-policy can be viewed as the special case of off-policy
- Off-policy has many additional uses in applications
  - Learn from data generated by a non-learning controller
  - Learn from a human expert
  - Learn multi-step predictive models of the world's dynamics

Q-Learning: Off-Policy TD Control

- We now consider off-policy learning of action-values $Q(s, a)$
- Next action is chosen using behavior policy $A_{t+1} \sim \mu(\cdot|S_t)$
- But we consider alternative successor action $A' \sim \pi(\cdot|S_t)$
- And update $Q(S_t, A_t)$ towards value of alternative action

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

Institute of Media, Information, and Network

Off-Policy Control with Q-Learning

- We now allow both behaviour and target policies to improve
- The target policy π is greedy w.r.t $Q(s, a)$

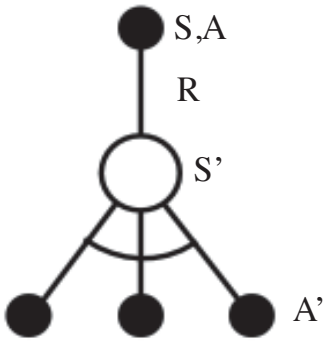
$$\pi(S_{t+1}) = \arg\max_{a'} Q(S_{t+1}, a')$$

- The behavior policy μ is e.g. ϵ-greedy w.r.t $Q(s, a)$
- The Q-learning target then simplifies:

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

Institute of Media, Information, and Network

Q-learning Control Algorithm



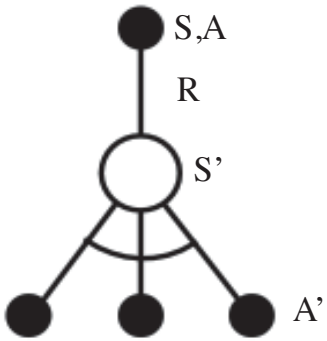
$$Q(S,A) \leftarrow Q(S,A) + \alpha\left(R + \gamma \max_{a'} Q(S',a') - Q(S,A)\right)$$

Theorem

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

Institute of Media, Information, and Network

Q-learning Control Algorithm (2)



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

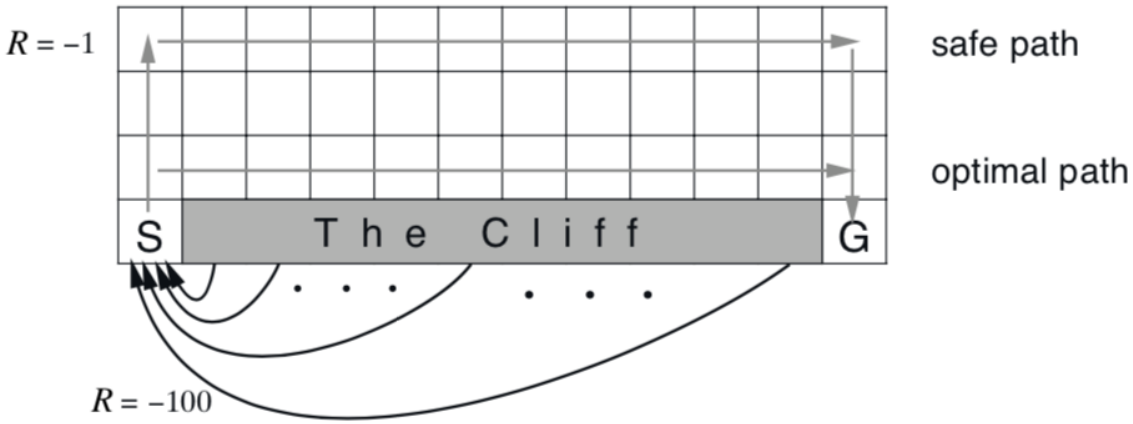
这样在状态 S_t 依据ε-greedy遵循策略得到的行为 A_t 的 Q 价值将朝着 S_{t+1} 状态所具有的最大 Q 价值的方向做一定比例的更新。这种算法能够使 $greedy$ 策略 π 最终收敛到最佳策略。由于个体实际与环境交互的时候遵循的是 $\epsilon - greedy$ 策略，它能保证经历足够丰富的新状态。

Institute of Media, Information, and Network

Q-Learning Algorithm for Off-Policy Control

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'
    $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$
    $S \leftarrow S'$;
  until S is terminal

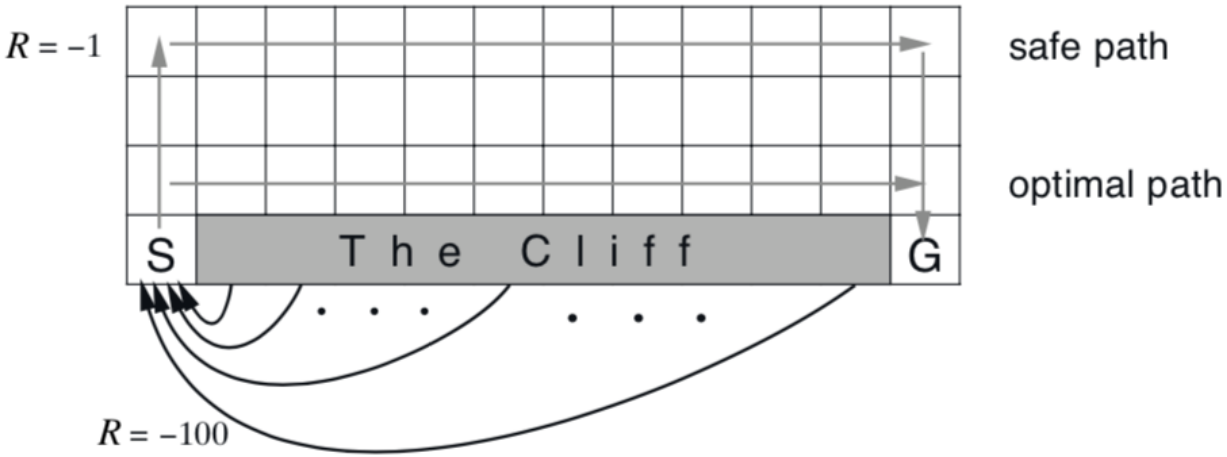
Cliff Walking Example

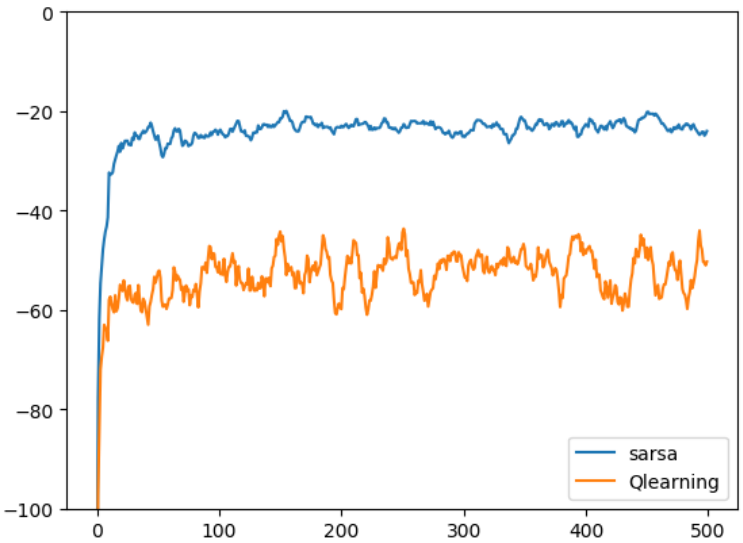


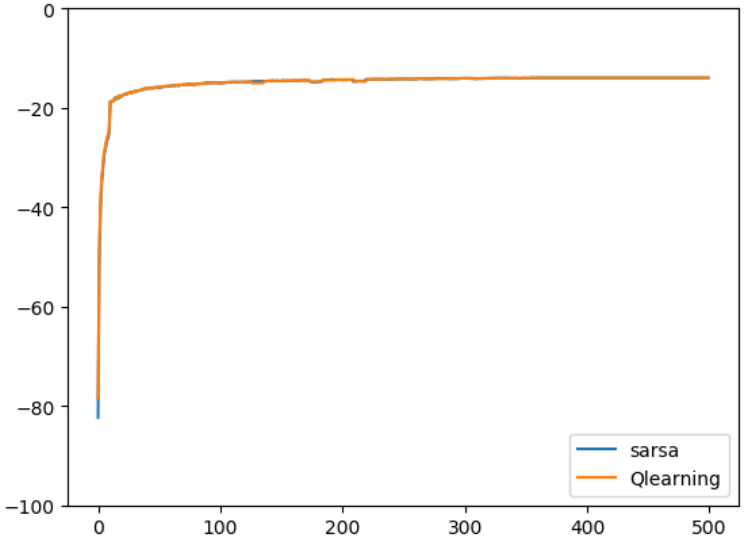
Example 6.6: Cliff Walking This gridworld example compares Sarsa and Q-learning, highlighting the difference between on-policy (Sarsa) and off-policy (Q-learning) methods. Consider the gridworld shown in the upper part of Figure 6.5. This is a standard undiscounted, episodic task, with start and goal states, and the usual actions causing movement up, down, right, and left. Reward is -1 on all transitions except those into the region marked "The Cliff." Stepping into this region incurs a reward of -100 and sends the agent instantly back to the start.

Institute of Media, Information, and Network

Cliff Walking Example (2)



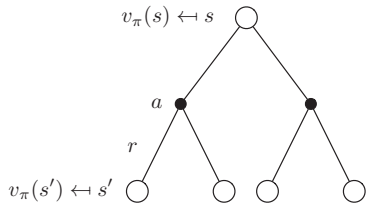
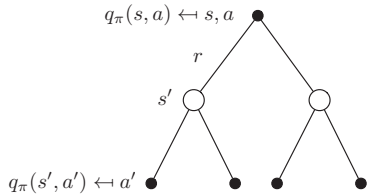
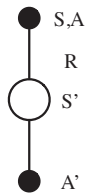
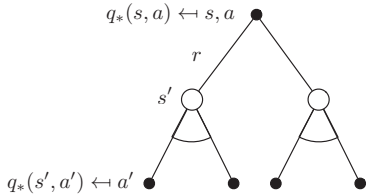




Media,
, and Network

Table of Contents

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

etwork

Relationship Between DP and TD (2)

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

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

Information, and Network