# Lecture 4: Model Free Control

Emma Brunskill

CS234 Reinforcement Learning.

Winter 2019

- Structure closely follows much of David Silver's Lecture 5. For additional reading please see SB Sections 5.2-5.4, 6.4, 6.5, 6.7

# Table of Contents

# Class Structure

- Last time: Policy evaluation with no knowledge of how the world works (MDP model not given)
- This time: Control (making decisions) without a model of how the world works
- Next time: Value function approximation

# Evaluation to Control

- Last time: how good is a specific policy?
  - Given no access to the decision process model parameters
  - Instead have to estimate from data / experience
- Today: how can we learn a good policy?

# Recall: Reinforcement Learning Involves

- Optimization
- Delayed consequences
- Exploration
- Generalization

# Today: Learning to Control Involves

- Optimization: Goal is to identify a policy with high expected rewards (similar to Lecture 2 on computing an optimal policy given decision process models)
- Delayed consequences: May take many time steps to evaluate whether an earlier decision was good or not
- Exploration: Necessary to try different actions to learn what actions can lead to high rewards

# Today: Model-free Control

- Generalized policy improvement
- Importance of exploration
- Monte Carlo control
- Model-free control with temporal difference (SARSA, Q-learning)
- Maximization bias

# Model-free Control Examples

- Many applications can be modeled as a MDP: Backgammon, Go, Robot locomation, Helicopter flight, Robocup soccer, Autonomous driving, Customer ad selection, Invasive species management, Patient treatment
- For many of these and other problems either:
    - MDP model is unknown but can be sampled
    - MDP model is known but it is computationally infeasible to use directly, except through sampling

# On and Off-Policy Learning

- On-policy learning
  - Direct experience
  - Learn to estimate and evaluate a policy from experience obtained from following that policy
- Off-policy learning
  - Learn to estimate and evaluate a policy using experience gathered from following a different policy

# Table of Contents

# Recall Policy Iteration

Random

$\pi(s) = a \ \forall s$

- Initialize policy $\pi$
- Repeat:
  - Policy evaluation: compute $V^\pi$
  - Policy improvement: update $\pi$

$$\pi'(s) = \arg\max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^\pi(s') = \arg\max_a Q^\pi(s,a)$$

- Now want to do the above two steps without access to the true dynamics and reward models
- Last lecture introduced methods for model-free policy evaluation

# Model Free Policy Iteration

- Initialize policy $\pi$
- Repeat:
  - Policy evaluation: compute $Q^\pi$ $\rightarrow$ $Q(s,a)$，直接利用Q进行改进.
  - Policy improvement: update $\pi$

# MC for On Policy Q Evaluation

Initialize $N(s,a) = 0,\ G(s,a) = 0,\ Q^{\pi}(s,a) = 0,\ \forall s \in S,\ \forall a \in A$
Loop

- Using policy $\pi$ sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$
- $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i - 1} r_{i,T_i}$
- For each **state,action** $(s,a)$ visited in episode $i$
  - For **first or every** time $t$ that $(s,a)$ is visited in episode $i$
    - $N(s,a) = N(s,a) + 1,\ G(s,a) = G(s,a) + G_{i,t}$
    - Update estimate $Q^{\pi}(s,a) = G(s,a)/N(s,a)$

# Model-free Generalized Policy Improvement

- Given an estimate $Q^{\pi_i}(s, a)\ \forall s, a$
- Update new policy

$$\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s, a) \tag{1}$$

# Model-free Policy Iteration

- Initialize policy $\pi$
- Repeat:
  - Policy evaluation: compute $Q^\pi$
  - Policy improvement: update $\pi$ given $Q^\pi$

- May need to modify policy evaluation:
  - If $\pi$ is deterministic, can't compute $Q(s, a)$ for any $a \neq \pi(s)$
- How to interleave policy evaluation and improvement?
  - Policy improvement is now using an estimated Q

# Table of Contents

# Policy Evaluation with Exploration

$Q^{\pi}(s, a)$

- Want to compute a model-free estimate of $Q^{\pi}$
- In general seems subtle
  - Need to try all $(s, a)$ pairs but then follow $\pi$
  - Want to ensure resulting estimate $Q^{\pi}$ is good enough so that policy improvement is a monotonic operator
- For certain classes of policies can ensure all (s,a) pairs are tried such that asymptotically $Q^{\pi}$ converges to the true value

# $\epsilon$-greedy Policies

- Simple idea to balance exploration and exploitation
- Let $|A|$ be the number of actions
- Then an $\epsilon$-greedy policy w.r.t. a state-action value $Q(s, a)$ is $\pi(a|s) = [\arg\max_a Q(s, a)$, w. prob $1 - \epsilon$; a w. prob $\frac{\epsilon}{|A|}]$

# Check Your Understanding: MC for On Policy Q Evaluation

Initialize $N(s,a) = 0,\ G(s,a) = 0,\ Q^\pi(s,a) = 0,\ \forall s \in S,\ \forall a \in A$
Loop

- Using policy $\pi$ sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$
- $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i - 1} r_{i,T_i}$
- For each **state,action** $(s,a)$ visited in episode $i$
  - For **first or every** time $t$ that $(s,a)$ is visited in episode $i$
    - $N(s,a) = N(s,a) + 1,\ G(s,a) = G(s,a) + G_{i,t}$
    - Update estimate $Q^\pi(s,a) = G(s,a)/N(s,a)$

- Mars rover with new actions:
  - $r(-,a_1) = [\ 1\ 0\ 0\ 0\ 0\ 0\ +10],\ r(-,a_2) = [\ 0\ 0\ 0\ 0\ 0\ 0\ +5],\ \gamma = 1.$
- Assume current greedy $\pi(s) = a_1\ \forall s$, $\epsilon$=.5
- Sample trajectory from $\epsilon$-greedy policy
- Trajectory = $(s_3, a_1, 0, s_2, a_2, 0, s_3, a_1, 0, s_2, a_2, 0, s_1, a_1, 1, \text{terminal})$
- First visit MC estimate of $Q$ of each $(s,a)$ pair? 合起来[1110000]
  $Q^{\epsilon-\pi}(-,a_1) = [1\ 0\ 1\ 0\ 0\ 0\ 0],\ Q^{\epsilon-\pi}(-,a_2) = [0\ 1\ 0\ 0\ 0\ 0\ 0]$

# Monotonic$\epsilon$-greedy Policy Improvement

## Theorem

For any $\epsilon$-greedy policy $\pi_i$, the $\epsilon$-greedy policy w.r.t. $Q^{\pi_i}$, $\pi_{i+1}$ is a monotonic improvement $V^{\pi_{i+1}} \geq V^{\pi}$

单调

$$
\begin{aligned}
Q^{\pi_i}(s, \pi_{i+1}(s)) &= \sum_{a \in A} \pi_{i+1}(a|s) Q^{\pi_i}(s, a) \\
&= (\epsilon/|A|) \sum_{a \in A} Q^{\pi_i}(s, a) + (1 - \epsilon) \max_a Q^{\pi_i}(s, a)
\end{aligned}
$$

- Therefore $V^{\pi_{i+1}} \geq V^{\pi}$ (from the policy improvement theorem)

# Monotonic $\epsilon$-greedy Policy Improvement

## Theorem

For any $\epsilon$-greedy policy $\pi_i$, the $\epsilon$-greedy policy w.r.t. $Q^{\pi_i}$, $\pi_{i+1}$ is a monotonic improvement $V^{\pi_{i+1}} \geq V^{\pi}$

$$
\begin{aligned}
Q^{\pi_i}(s, \pi_{i+1}(s)) &= \sum_{a \in A} \pi_{i+1}(a|s) Q^{\pi_i}(s,a) \\
&= (\epsilon/|A|) \sum_{a \in A} Q^{\pi_i}(s,a) + (1-\epsilon) \max_a Q^{\pi_i}(s,a) \\
&= (\epsilon/|A|) \sum_{a \in A} Q^{\pi_i}(s,a) + (1-\epsilon) \max_a Q^{\pi_i}(s,a) \frac{1-\epsilon}{1-\epsilon} \\
&= (\epsilon/|A|) \sum_{a \in A} Q^{\pi_i}(s,a) + (1-\epsilon) \max_a Q^{\pi_i}(s,a) \sum_{a \in A} \frac{\pi_i(a|s) - \frac{\epsilon}{|A|}}{1-\epsilon} \\
&\geq \frac{\epsilon}{|A|} \sum_{a \in A} Q^{\pi_i}(s,a) + (1-\epsilon) \sum_{a \in A} \frac{\pi_i(a|s) - \frac{\epsilon}{|A|}}{1-\epsilon} Q^{\pi_i}(s,a) \\
&= \sum_{a \in A} \pi_i(a|s) Q^{\pi_i}(s,a) = V^{\pi_i}(s)
\end{aligned}
$$

- Therefore $V^{\pi_{i+1}} \geq V^{\pi}$ (from the policy improvement theorem)

# Subtleties of Policy Improvement

- Note that when we first introduced policy improvement with a given MDP dynamics and reward model, policy evaluation was computed *exactly*.
- In this case monotonic improvement was guaranteed for each policy improvement step.
- In this lecture we will often be considering computing a $Q$ using samples gathered from many policies
- Beautifully, generalized policy iteration using MC and TD methods still converge under some mild conditions
- For more technical details, proofs of the convergence of Q-learning for different scenarios can be found here:
  - Q-Learning. Watkins and Dayan. *Machine Learning*. 1992
  - Asynchronous Stochastic Approximation and Q-Learning. Tsitsiklis. *Machine Learning*. 1994

# Greedy in the Limit of Infinite Exploration (GLIE)

## Definition of GLIE

- All state-action pairs are visited an infinite number of times

$$\lim_{i \to \infty} N_i(s, a) \to \infty$$

- Behavior policy converges to greedy policy
  $\lim_{i \to \infty} \pi(a|s) \to \arg\max_a Q(s, a)$ with probability 1

- A simple GLIE strategy is $\epsilon$-greedy where $\epsilon$ is reduced to 0 with the following rate: $\epsilon_i = 1/i$

# Table of Contents

# Monte Carlo Online Control / On Policy Improvement

Initialize $Q(s,a) = 0, N(s,a) = 0 \ \forall (s,a)$, Set $\epsilon = 1$, $k = 1$
$\pi_k = \epsilon\text{-greedy}(Q)$ // Create initial $\epsilon$-greedy policy
**loop**
    Sample $k$-th episode $(s_{k,1}, a_{k,1}, r_{k,1}, s_{k,2}, \ldots, s_{k,T})$ given $\pi_k$
    $G_{k,t} = r_{k,t} + \gamma r_{k,t+1} + \gamma^2 r_{k,t+2} + \cdots \gamma^{T_i - 1} r_{k,T_i}$
    **for** $t = 1, \ldots, T$ **do**
        **if** First visit to $(s,a)$ in episode $k$ **then**
            $N(s,a) = N(s,a) + 1$
            $Q(s_t, a_t) = Q(s_t, a_t) + \frac{1}{N(s,a)}(G_{k,t} - Q(s_t, a_t))$
        **end if**
    **end for**
    $k = k + 1$, $\epsilon = 1/k$
    $\pi_k = \epsilon\text{-greedy}(Q)$ // Policy improvement
**end loop**

# Check Your Understanding: MC for On Policy Control

- Mars rover with new actions:
  - $r(-, a_1) = [\ 1\ 0\ 0\ 0\ 0\ 0\ {+10}]$, $r(-, a_2) = [\ 0\ 0\ 0\ 0\ 0\ 0\ {+5}]$, $\gamma = 1$.
- Assume current greedy $\pi(s) = a_1\ \forall s$, $\epsilon$=.5
- Sample trajectory from $\epsilon$-greedy policy
- Trajectory = ($s_3$, $a_1$, 0, $s_2$, $a_2$, 0, $s_3$, $a_1$, 0, $s_2$, $a_2$, 0, $s_1$, $a_1$, 1, terminal)
- First visit MC estimate of $Q$ of each $(s, a)$ pair?
- $Q^{\epsilon-\pi}(-, a_1) = [1\ 0\ 1\ 0\ 0\ 0\ 0]$, $Q^{\epsilon-\pi}(-, a_2) = [0\ 1\ 0\ 0\ 0\ 0\ 0]$
- What is $\pi(s) = \arg\max_a Q^{\epsilon-\pi}(s, a)\ \forall s$?
  $\pi$ = [1 2 1 tie tie tie tie]
- What is new $\epsilon$-greedy policy, if $k = 3$, $\epsilon = 1/k$
  With probability 2/3 choose $\pi(s)$ else choose randomly

# GLIE Monte-Carlo Control

**Theorem**

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

# Model-free Policy Iteration

- Initialize policy $\pi$
- Repeat:
    - Policy evaluation: compute $Q^\pi$
    - Policy improvement: update $\pi$ given $Q^\pi$

- What about TD methods?

bootstrapping.(自举)

$$V^\pi(s) = V^\pi(s) + \alpha(\underbrace{r + \gamma V^\pi(s')}_{\text{Sampling expectation.}} - V^\pi(s))$$

# Table of Contents

# Model-free Policy Iteration with TD Methods

- Use temporal difference methods for policy evaluation step
- Initialize policy $\pi$
- Repeat:
    - Policy evaluation: compute $Q^{\pi}$ using temporal difference updating with $\epsilon$-greedy policy
    - Policy improvement: Same as Monte carlo policy improvement, set $\pi$ to $\epsilon$-greedy $(Q^{\pi})$

# General Form of SARSA Algorithm

Set initial $\epsilon$-greedy policy $\pi$ randomly, $t = 0$, initial state $s_t = s_0$
Take $a_t \sim \pi(s_t)$ // Sample action from policy
Observe $(r_t, s_{t+1})$
**loop**
    Take action $a_{t+1} \sim \pi(s_{t+1})$
    Observe $(r_{t+1}, s_{t+2})$
    Update Q given $(s_t, a_t, r_t, s_{t+1}, a_{t+1})$:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + ((1-\alpha)(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)))$$

    Perform policy improvement:

    $t = t + 1$
**end loop**

# General Form of SARSA Algorithm

Set initial $\epsilon$-greedy policy $\pi$, $t = 0$, initial state $s_t = s_0$
Take $a_t \sim \pi(s_t)$ // Sample action from policy
Observe $(r_t, s_{t+1})$
**loop**
    Take action $a_{t+1} \sim \pi(s_{t+1})$
    Observe $(r_{t+1}, s_{t+2})$
    $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$
    $\pi(s_t) = \arg\max_a Q(s_t, a)$ w.prob $1 - \epsilon$, else random
    $t = t + 1$
**end loop**

What are the benefits to improving the policy after each step?
What are the benefits to updating the policy less frequently?

# Convergence Properties of SARSA

## Theorem

SARSA for finite-state and finite-action MDPs converges to the optimal action-value, $Q(s, a) \rightarrow Q^*(s, a)$, under the following conditions:

1. The policy sequence $\pi_t(a|s)$ satisfies the condition of GLIE
2. The step-sizes $\alpha_t$ satisfy the Robbins-Munro sequence such that

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

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

# Convergence Properties of SARSA

**Theorem**

SARSA for finite-state and finite-action MDPs converges to the optimal action-value, $Q(s,a) \to Q^*(s,a)$, under the following conditions:

1. The policy sequence $\pi_t(a|s)$ satisfies the condition of GLIE
2. The step-sizes $\alpha_t$ satisfy the Robbins-Munro sequence such that

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

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

Would one want to use a step size choice that satisfies the above in practice? Likely not.

# Q-Learning: Learning the Optimal State-Action Value

- Can we estimate the value of the optimal policy $\pi^*$ without knowledge of what $\pi^*$ is?
- Yes! Q-learning
- Key idea: Maintain state-action $Q$ estimates and use to bootstrap– use the value of the best future action
- Recall SARSA

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha((r_t + \gamma Q(s_{t+1}, a_{t+1})) - Q(s_t, a_t)) \tag{2}$$

- Q-learning:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha((r_t + \gamma \max_{a'} Q(s_{t+1}, a')) - Q(s_t, a_t)) \tag{3}$$

# Off-Policy Control Using Q-learning

- In the prior slide assumed there was some $\pi_b$ used to act
- $\pi_b$ determines the actual rewards received
- Now consider how to improve the behavior policy (policy improvement)
- Let behavior policy $\pi_b$ be $\epsilon$-greedy with respect to (w.r.t.) current estimate of the optimal $Q(s, a)$

# Q-Learning with $\epsilon$-greedy Exploration

```
Initialize Q(s, a), ∀s ∈ S, a ∈ A  t = 0, initial state s_t = s_0
Set π_b to be ε-greedy w.r.t. Q
loop
    Take a_t ∼ π_b(s_t) // Sample action from policy
    Observe (r_t, s_{t+1})
    Update Q given (s_t, a_t, r_t, s_{t+1}):


    Perform policy improvement: set π_b to be ε-greedy w.r.t. Q
    t = t + 1
end loop
```

# Q-Learning with $\epsilon$-greedy Exploration

```
Initialize Q(s, a), ∀s ∈ S, a ∈ A  t = 0, initial state s_t = s_0
Set π_b to be ε-greedy w.r.t. Q
loop
    Take a_t ∼ π_b(s_t) // Sample action from policy
    Observe (r_t, s_{t+1})
    Q(s_t, a_t) ← Q(s_t, a_t) + α(r_t + γ arg max_a Q(s_{t_1}, a) − Q(s_t, a_t))
    π(s_t) = arg max_a Q(s_t, a) w.prob 1 − ε, else random
    t = t + 1
end loop
```

Does how $Q$ is initialized matter?

Asymptotically no, under mild condiditions, but at the beginning, yes

# Check Your Understanding: Q-learning

- Mars rover with new actions:
  - $r(-,a_1) = [\ 1\ 0\ 0\ 0\ 0\ 0\ +10]$, $r(-,a_2) = [\ 0\ 0\ 0\ 0\ 0\ 0\ +5]$, $\gamma = 1$.
- Assume current greedy $\pi(s) = a_1\ \forall s$, $\epsilon$=.5
- Sample trajectory from $\epsilon$-greedy policy
- Trajectory = ($s_3$, $a_1$, 0, $s_2$, $a_2$, 0, $s_3$, $a_1$, 0, $s_2$, $a_2$, 0, $s_1$, $a_1$, 1, terminal)
- New $\epsilon$-greedy policy under MC, if $k = 3$, $\epsilon = 1/k$: with probability 2/3 choose $\pi$ = [1 2 1 tie tie tie tie], else choose randomly
- Q-learning updates? Initialize $\epsilon = 1/k$, $k = 1$, and $\alpha = 0.5$
- $\pi$ is random with probability $\epsilon$, else $\pi = [\ 1\ 1\ 1\ 2\ 1\ 2\ 1]$
- First tuple: $(s_3, a_1, 0, s_2)$.
- Q-learning:
  $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha(r_t + \gamma \arg\max_a Q(s_{t_1},a) - Q(s_t,a_t))$
  Update $Q(s_3,a_1) = 0$. $k = 2$

  New policy is random with probability $1/k$ else
  $\pi(s) = \arg\max Q(s_3,a)$ = tie between actions 1 and 2.

# Q-Learning with $\epsilon$-greedy Exploration

- What conditions are sufficient to ensure that Q-learning with $\epsilon$-greedy exploration converges to optimal $Q^*$?
  Visit all $(s, a)$ pairs infinitely often, and the step-sizes $\alpha_t$ satisfy the Robbins-Munro sequence. Note: the algorithm does not have to be greedy in the limit of infinite exploration (GLIE) to satisfy this (could keep $\epsilon$ large).

- What conditions are sufficient to ensure that Q-learning with $\epsilon$-greedy exploration converges to optimal $\pi^*$?
  The algorithm is GLIE, along with the above requirement to ensure the Q value estimates converge to the optimal Q.

# Table of Contents

# Maximization Bias[1]

- Consider single-state MDP ($|S| = 1$) with 2 actions, and both actions have 0-mean random rewards, ($\mathbb{E}(r|a = a_1) = \mathbb{E}(r|a = a_2) = 0$).
- Then $Q(s, a_1) = Q(s, a_2) = 0 = V(s)$
- Assume there are prior samples of taking action $a_1$ and $a_2$
- Let $\hat{Q}(s, a_1), \hat{Q}(s, a_2)$ be the finite sample estimate of $Q$
- Use an unbiased estimator for $Q$: e.g. $\hat{Q}(s, a_1) = \frac{1}{n(s,a_1)} \sum_{i=1}^{n(s,a_1)} r_i(s, a_1)$
- Let $\hat{\pi} = \arg\max_a \hat{Q}(s, a)$ be the greedy policy w.r.t. the estimated $\hat{Q}$
- *Even though each estimate of the state-action values is unbiased*, the estimate of $\hat{\pi}$'s value $\hat{V}^{\hat{\pi}}$ can be biased:
  $\hat{V}^{\hat{\pi}}(s) = \mathbb{E}[\max \hat{Q}(s, a_1), \hat{Q}(s, a_2)]$
  $\geq \max[\mathbb{E}[\hat{Q}(s, a_1)], [\hat{Q}(s, a_2)]]$
  $= max[0, 0] = V^{\pi}$,
  where the inequality comes from Jensen's inequality.

[1]Example from Mannor, Simester, Sun and Tsitsiklis. Bias and Variance Approximation in Value Function Estimates. Management Science 2007

# Double Learning

- The greedy policy w.r.t. estimated $Q$ values can yield a maximization bias during finite-sample learning
- Avoid using max of estimates as estimate of max of true values
- Instead split samples and use to create two independent unbiased estimates of $Q_1(s_1, a_i)$ and $Q_2(s_1, a_i)$ $\forall a$.
  - Use one estimate to select max action: $a^* = \arg\max_a Q_1(s_1, a)$
  - Use other estimate to estimate value of $a^*$: $Q_2(s, a^*)$
  - Yields unbiased estimate: $\mathbb{E}(Q_2(s, a^*)) = Q(s, a^*)$
- Why does this yield an unbiased estimate of the max state-action value?

  Using independent samples to estimate the value

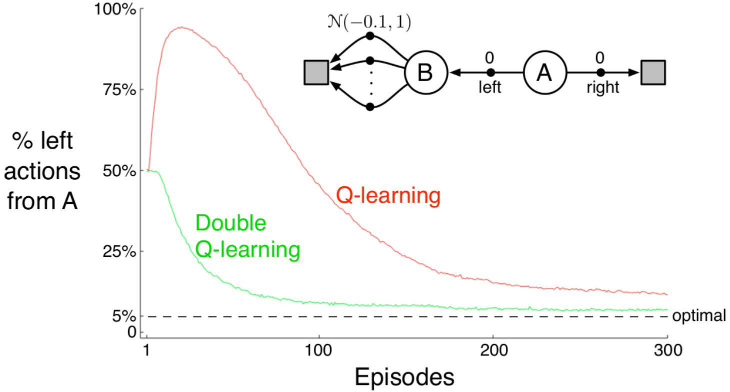
- If acting online, can alternate samples used to update $Q_1$ and $Q_2$, using the other to select the action chosen
- Next slides extend to full MDP case (with more than 1 state)

# Double Q-Learning

Initialize $Q_1(s,a)$ and $Q_2(s,a)$,$\forall s \in S, a \in A$ $t=0$, initial state $s_t = s_0$
**loop**
  Select $a_t$ using $\epsilon$-greedy $\pi(s) = \arg\max_a Q_1(s_t,a) + Q_2(s_t,a)$
  Observe $(r_t, s_{t+1})$
  **if** (with 0.5 probability) **then**
    $Q_1(s_t,a_t) \leftarrow Q_1(s_t,a_t) + \alpha(r_t + \gamma \max_a Q_2(s_{t+1},a)$
  **else**
    $Q_2(s_t,a_t) \leftarrow Q_2(s_t,a_t) + \alpha(r_t + \gamma \max_a Q_1(s_{t+1},a)$
  **end if**
  $t = t+1$
**end loop**

- Compared to Q-learning, how does this change the: memory requirements, computation requirements per step, amount of data required?
  Doubles the memory, same computation requirements, data requirements are subtle– might reduce amount of exploration needed due to lower bias

# Double Q-Learning (Figure 6.7 in Sutton and Barto 2018)




Due to the maximization bias, Q-learning spends much more time selecting suboptimal actions than double Q-learning.

# Table of Contents

# What You Should Know

- Be able to implement MC on policy control and SARSA and Q-learning
- Compare them according to properties of how quickly they update, (informally) bias and variance, computational cost
- Define conditions for these algorithms to converge to the optimal Q and optimal $\pi$ and give at least one way to guarantee such conditions are met.

# Class Structure

- Last time: Policy evaluation with no knowledge of how the world works (MDP model not given)
- This time: Control (making decisions) without a model of how the world works
- **Next time: Value function approximation**