清華大學
Tsinghua University

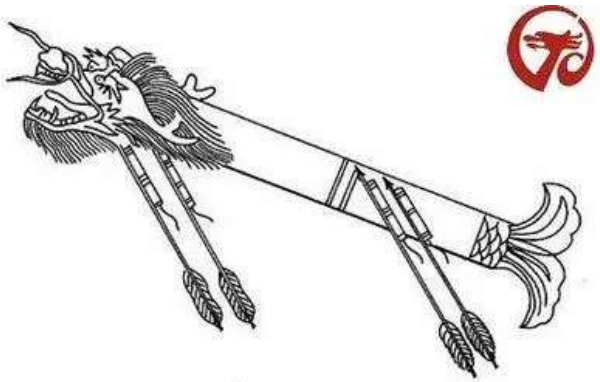
《强化学习与控制》
--
Monte Carlo RL

Shengbo Eben Li
(李升波)

Intelligent Driving Laboratory (*i*DLab)

Tsinghua University

Success is a pile of failure



Progress is made by trial and failure;
the failures are generally a hundred times
more numerous than the successes;
yet they are usually left unchronicled.

-- William Ramsay (1852-1916)

Outline

1. Monte Carlo Estimation
2. MC Algorithm Design
3. On-Policy vs Off-policy
4. Broad Viewpoint of MC

Monte Carlo Estimation

Monte Carlo method

- First developed by Stanislaw Ulam in the late 1940s (Polish-American mathematician)
- Worked in Manhattan Project to carry out hydro-dynamical calculations
- Being secret, MC method needed a code name: Monte Carlo casino in Monaco was where Ulam's uncle liked to gamble

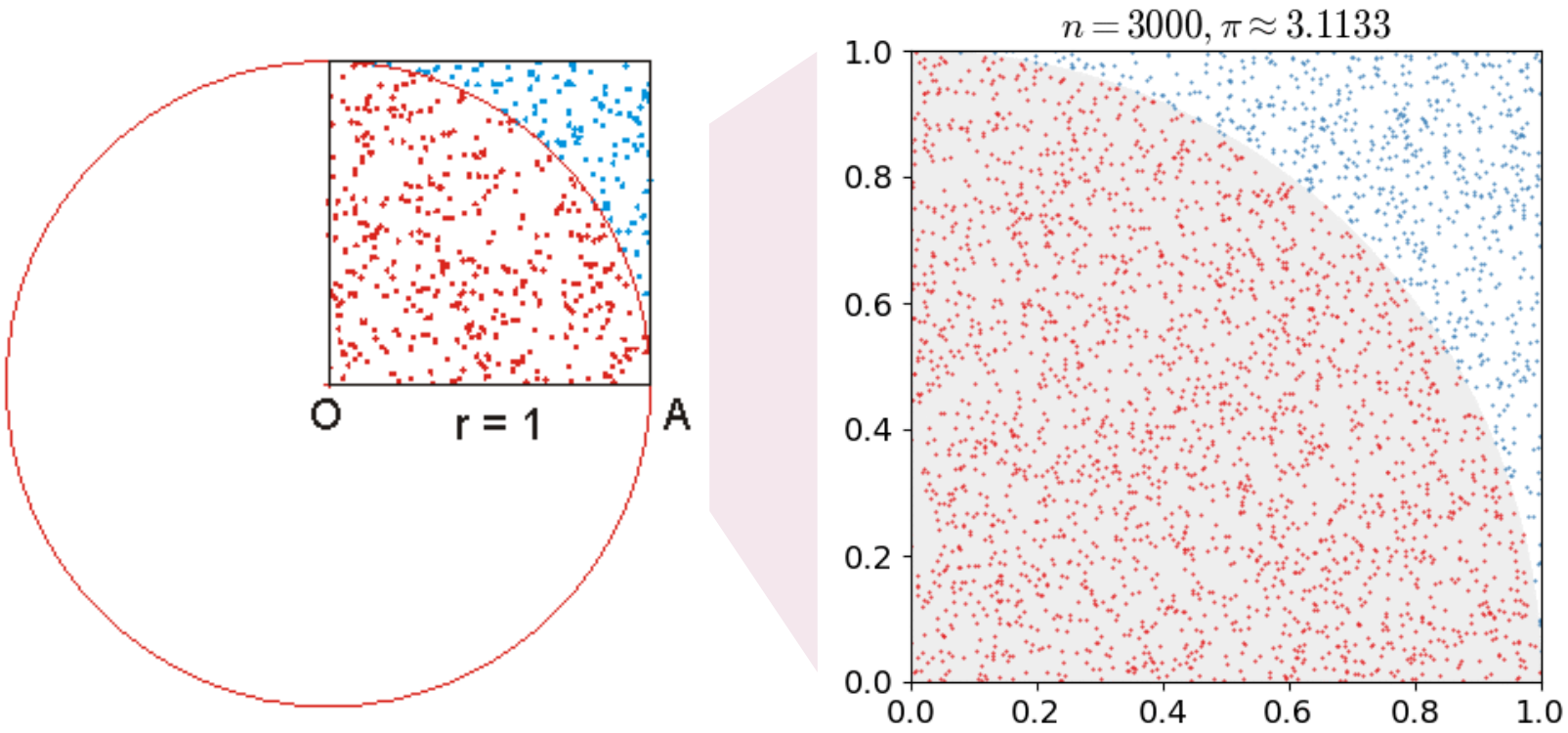
Stanislaw Ulam

Monte Carlo casino

Monte Carlo Estimation

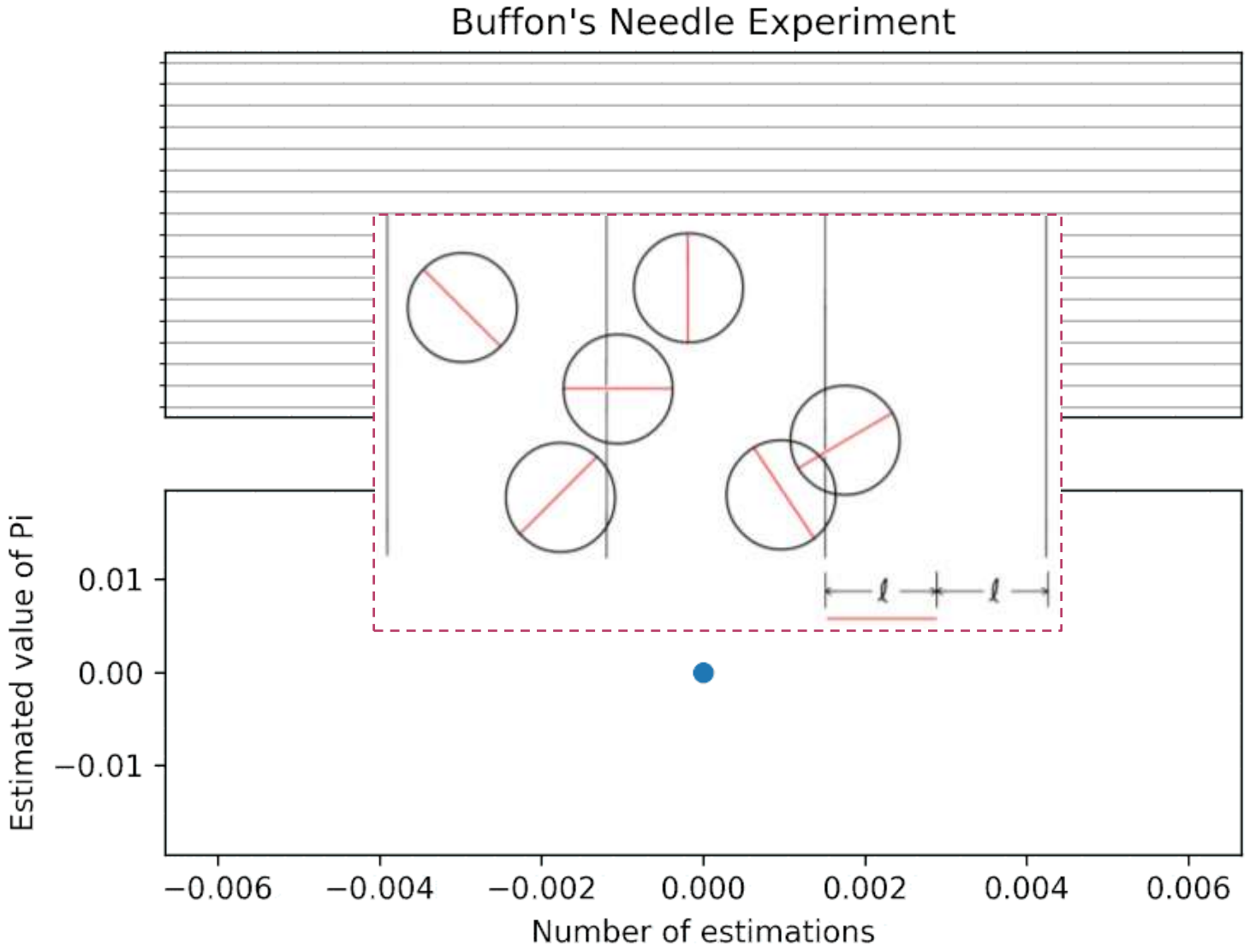
Monte Carlo experiment

- The key idea is to use repeated sampling to calculate the statistical properties of some random phenomena or behaviors



Monte Carlo Estimation

- **Buffon's needle problem**



Monte Carlo Estimation

- **Monte Carlo (MC) estimation is the simplest sample-based estimation**

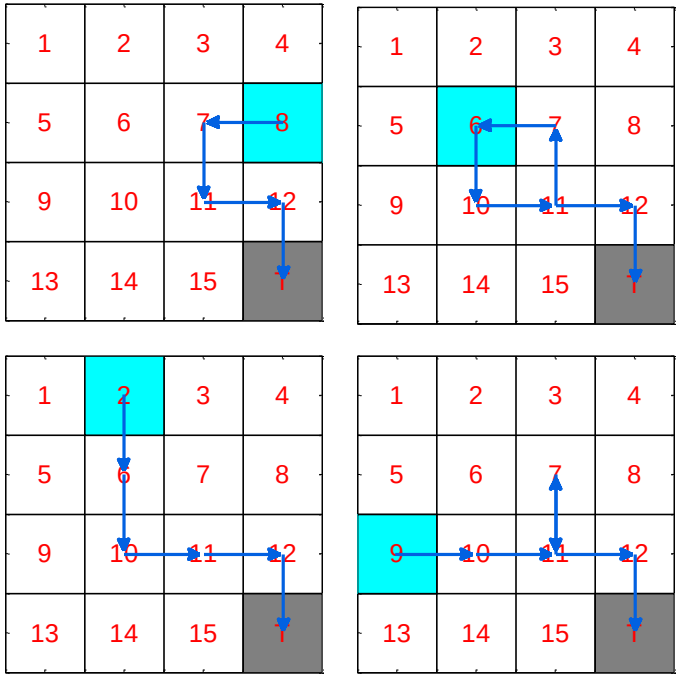
- **Featured with**

  - Model-free: no knowledge of environment model
  - Learn from experience by interacting with environment
  - Use a complete trajectory and no bootstrapping
  - Only suits episodic tasks, i.e., tasks must terminate

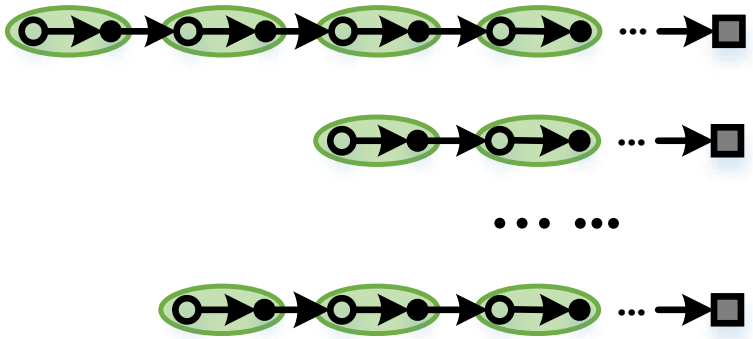
Monte Carlo Estimation

Basic idea of MC estimation

- **Evaluate value function by calculating average return from a collection of episodes**



Generate a few episodes by interacting with environment



Monte Carlo Estimation

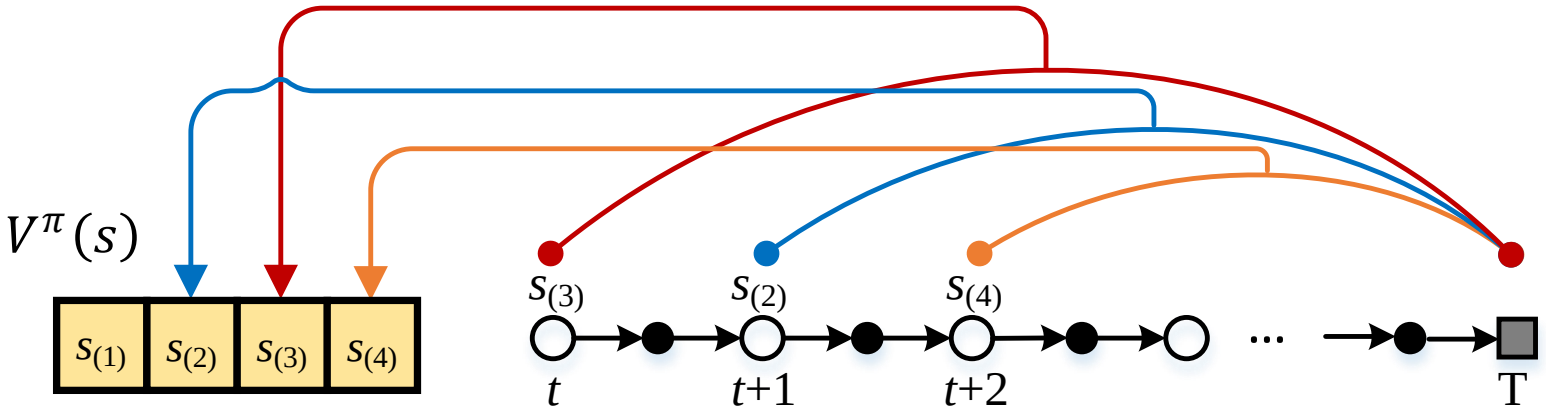
Basic idea of MC estimation

- Estimate of state-value function $V^{\pi}(s)$

$$V^{\pi}(s) = \text{Avg}\{G_{t:T} | s_t = s\}$$

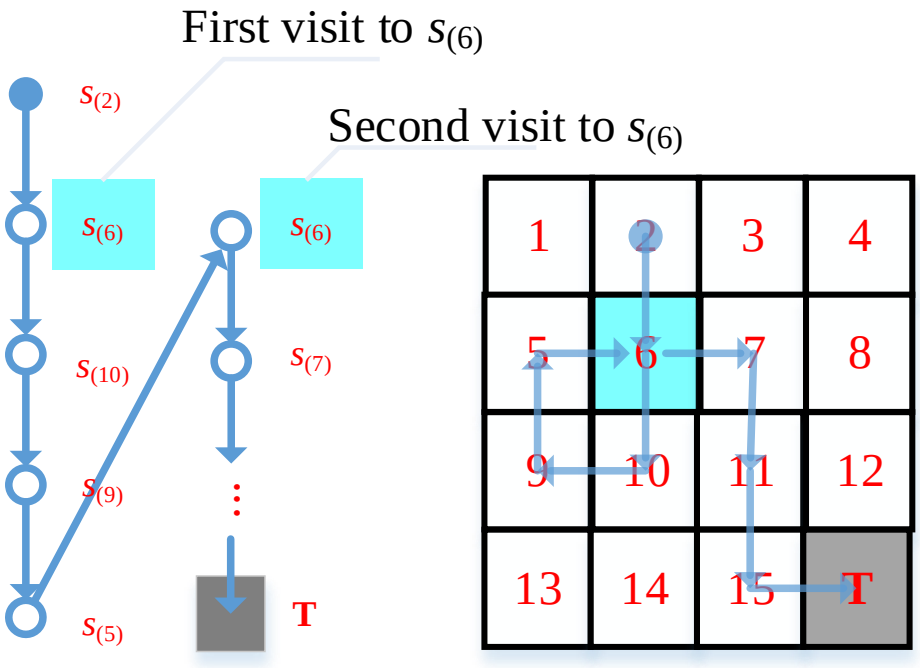
$$G_{t:T} = \sum_{i=0}^{T-t} \gamma^i r_{t+i}$$

Average all the returns followed after a particular state s



(a) Estimation of state-value function

First Visit MC VS Every Visit MC

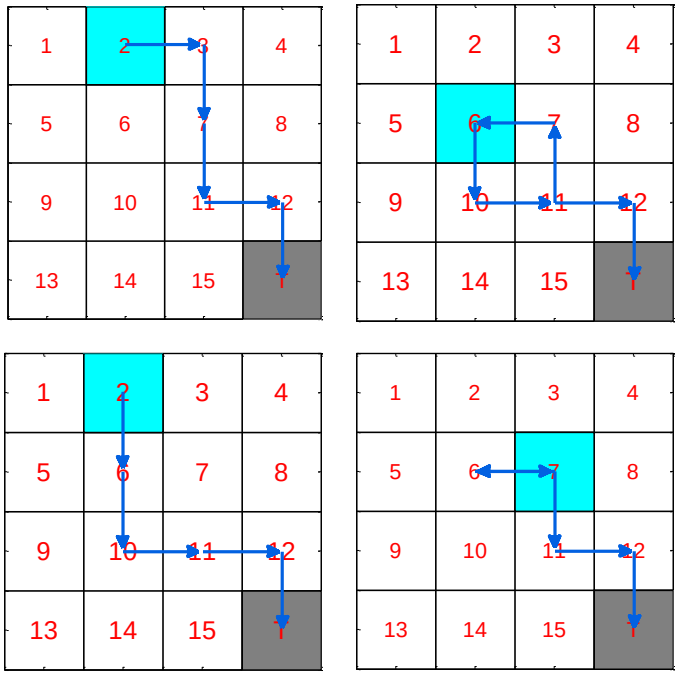


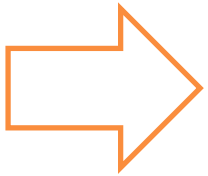
How many states can be updated with MC estimation?

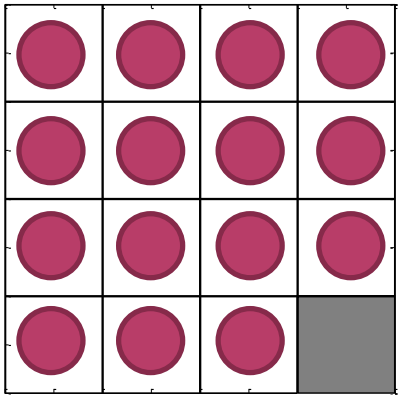
- First-visit MC: estimate as the average of returns only following first visits to s
- Every-visit MC: estimate as the average of returns following all visits to s

Monte Carlo estimation

- **An example: Given a policy, 4 episodes are collected**







? $V^{\pi}(s_{(7)})$

$V^{\pi}(s_{(10)})$

$V^{\pi}(s_{(5)})$

Starting state

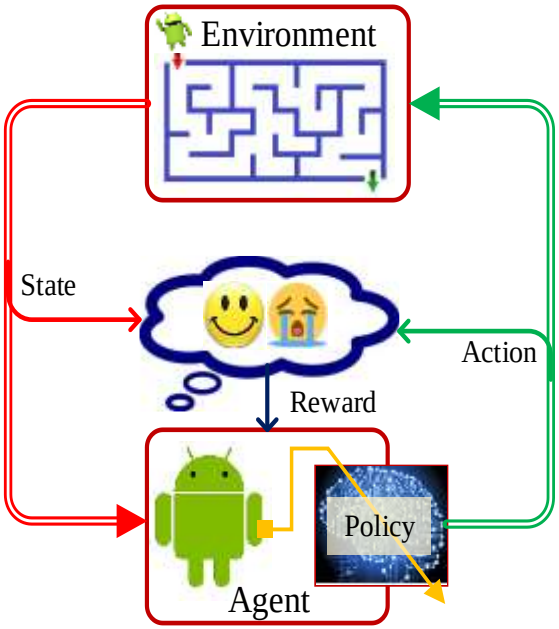
$$r(s,a,s') = \begin{cases} -1 & \text{if } s' \neq \text{T} \\ +9\text{'} & \text{if } s' = \text{T} \end{cases}$$

Discount factor: $\gamma = 0.9$

Outline

1. Monte Carlo Estimation
2. MC Algorithm Design
3. On-Policy vs Off-policy
4. Broad Viewpoint of MC

Model-free RL Problem



$$\max_{\pi} J(\pi) = \mathbb{E}_{s\sim d_{\text{init}}(s)}\{v^{\pi}(s)\}$$

Subject to:

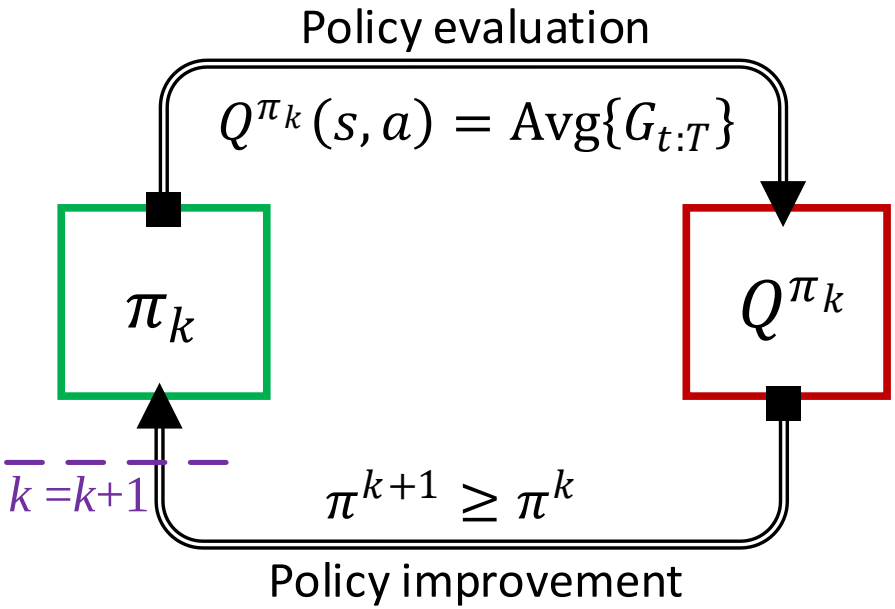
$$\mathcal{D} = \left\{ \begin{array}{c} s_0, a_0, \\ s_1, a_1, \\ s_2, a_2, \\ \cdots\cdots, \\ s_t, a_t, \\ s_{t+1}, a_{t+1}, \\ \cdots \end{array} \right\}$$

(1) Environment model
(2) State-action samples
(3) Policy
(4) Reward (value function)

Optimal policy: π^* ?

Monte Carlo learning algorithm

- (1) Policy EValuation (**PEV**) and (2) Policy IMprovement (**PIM**)



- **(1) PEV**: apply Monte Carlo estimation to approximate the true action-value function of the current policy
- **(2) PIM**: find a new better policy based on the estimate of action-value function

MC Policy Evaluation

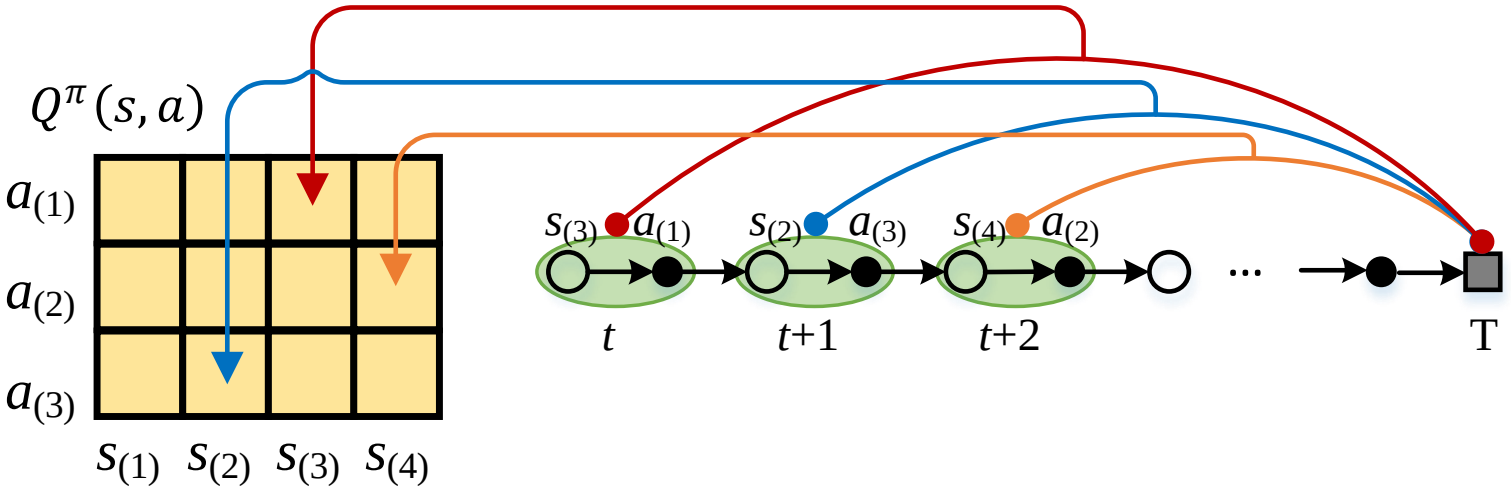
Policy EValuation (PEV)

- MC estimate of action-value function $Q^{\pi}(s,a)$

$$Q^{\pi}(s,a) = \text{Avg}\{G_{t:T} | s_t = s, a_t = a\}$$

$$G_{t:T} = \sum_{i=0}^{T-t} \gamma^i r_{t+i}$$

Average all the returns followed after a particular state-action pair (s,a)

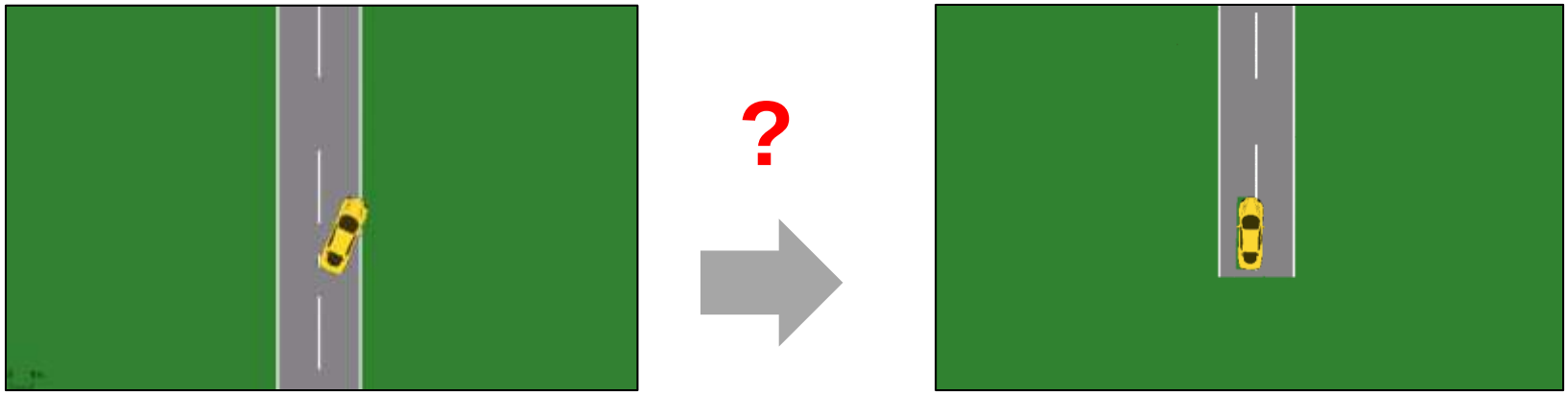


(b) Estimation of action-value function

MC Policy Improvement

Policy IMprovement (PIM)

- What is a better policy?



Bad policy

Good policy

- A policy $\bar{\pi}$ is said to be better than π if each element of its state-value function is larger than that of its counterpart

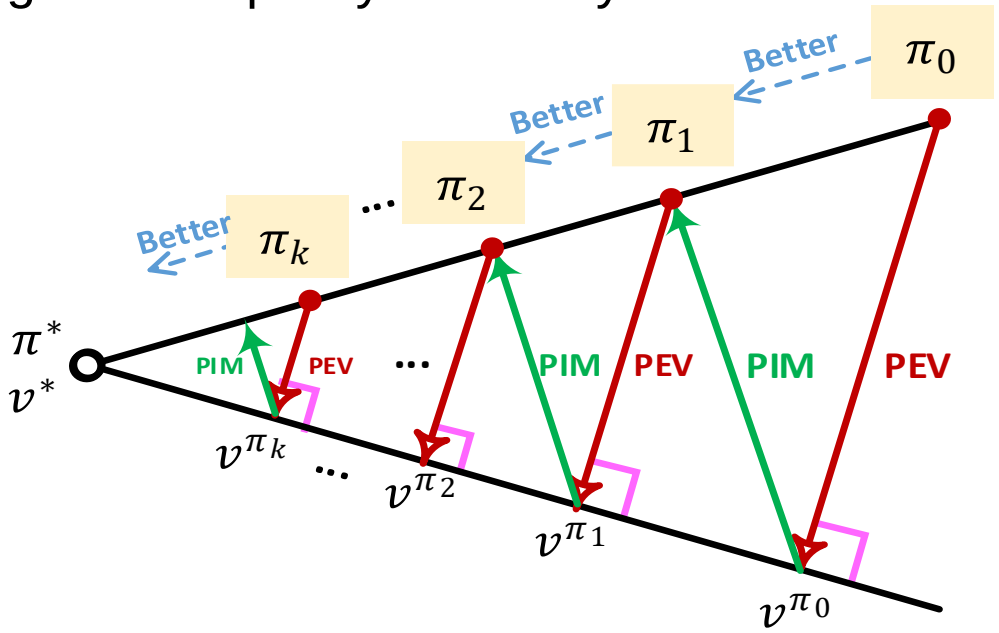
$$\pi \leq \bar{\pi} \Leftrightarrow v^{\pi}(s) \leq v^{\bar{\pi}}(s), \forall s \in \mathcal{S}$$

Element-by-element definition

MC Policy Improvement

Policy IMprovement (PIM)

- Finding a better policy is the key to ensure convergence



- A sufficient condition: **Policy improvement theorem**

A policy $\bar{\pi}$ is better than π if

$$v^{\pi}(s) \leq \sum_{a \in \mathcal{A}} \bar{\pi}(a|s) q^{\pi}(s,a), \forall s \in \mathcal{S}$$

(Sutton & Barto, 2018)

MC Policy Improvement

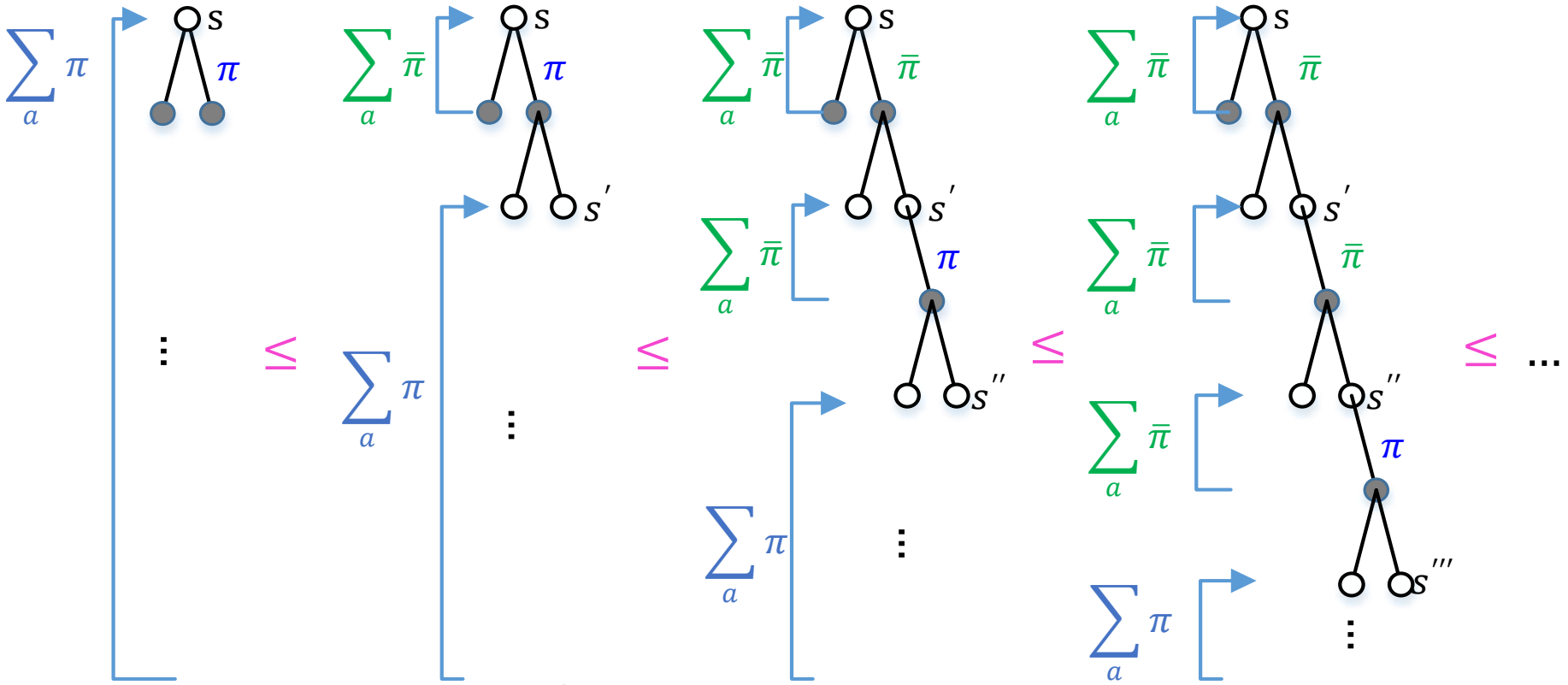
Proof of policy improvement theorem

- Cascading inequality by replacing π with $\bar{\pi}$

$$v^{\pi}(s) \le \sum_{a \in \mathcal{A}} \bar{\pi}(a|s) q^{\pi}(s,a)$$

$$v^{\pi}(s') \le \sum_{a \in \mathcal{A}} \bar{\pi}(a'|s') q^{\pi}(s',a')$$

$$v^{\pi}(s'') \le \sum_{a \in \mathcal{A}} \bar{\pi}(a''|s'') q^{\pi}(s'',a'')$$



MC Policy Improvement

Proof of policy improvement theorem

- Cascading inequality by replacing π with $\bar{\pi}$

$$
\begin{aligned}
v^{\pi}(s) &\le \sum_{a\in\mathcal{A}} \bar{\pi}(a|s) q^{\pi}(s,a) \\
&= \mathbb{E}_{a\sim\bar{\pi}}\{q^{\pi}(s,a)\} \\
&= \mathbb{E}_{a\sim\bar{\pi}}\left\{\sum_{s'\in S} \mathcal{P}_{ss'}^{a}\left(r+\gamma v^{\pi}(s')\right)\right\} \\
&= \mathbb{E}_{a,s'\sim\bar{\pi}}\{r+\gamma v^{\pi}(s')\} \\
&\le \mathbb{E}_{a,s'\sim\bar{\pi}}\left\{r+\gamma \mathbb{E}_{a',s''\sim\bar{\pi}}\{r'+v^{\pi}(s'')\}\right\} \\
&= \mathbb{E}_{a,s',a',s''\sim\bar{\pi}}\{r+\gamma r'+\gamma^2 v^{\pi}(s'')\}
\end{aligned}
$$

Roll forward until $t \to +\infty$ to obtain:

$$
\begin{aligned}
&\le \mathbb{E}_{a,s',a',s'',a'',s''',\cdots\sim\bar{\pi}}\{r+\gamma r'+\gamma^2 r''+\cdots+\gamma^{\infty} r_{\infty}\} \\
&= v^{\bar{\pi}}(s)
\end{aligned}
$$

MC Policy Improvement

Two popular choices

- (1) greedy policy $\pi^g(a|s) = \begin{cases} 1, & \text{if } a = a^* \\ 0, & \text{if } a \neq a^* \end{cases}$

  where $a^* = \arg\max_a q^\pi(s,a)$

  Action-value function

- (2) ϵ-greedy policy

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

MC Policy Improvement

(1) Greedy policy

- The key is to satisfy policy improvement theorem

$$v^{\pi^g}(s) \leq \sum_{a \in \mathcal{A}} \pi^g_{\text{new}}(a|s) q^{\pi^g}(s,a), \forall s \in \mathcal{S}$$

Proof:

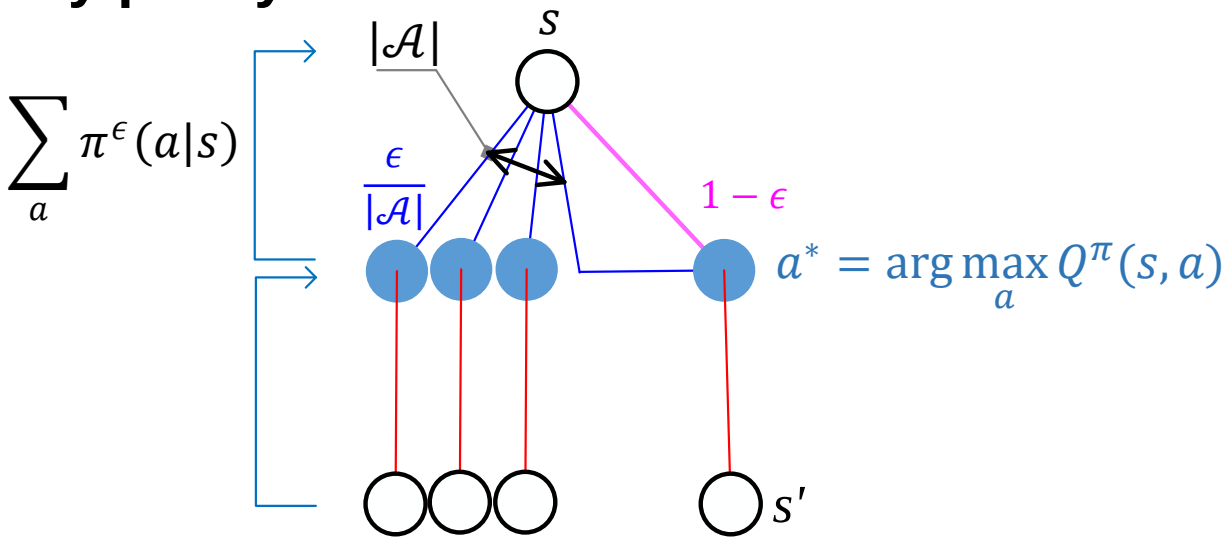
$$\begin{aligned} v^{\pi^g}(s) &= q^{\pi^g}\left(s, \pi^g(s)\right) \\ &\leq \max_a q^{\pi^g}(s,a) \\ &= q^{\pi^g}\left(s, \arg\max_a q^{\pi^g}(s,a)\right) \\ &= q^{\pi^g}\left(s, \pi^g_{\text{new}}(s)\right) \end{aligned}$$

where $\pi^g(a|s)$ is greedy policy, k is cycle step

- Note that the inequality holds because a^* is the best action among all actions

MC Policy Improvement

(2) ϵ-greedy policy



- (1) Random part: Select random action in a small probability of ϵ (each action with $\epsilon/|\mathcal{A}|$)
- (2) Greedy part: Select the best action in the probability of $1-\epsilon$

- This policy can NOT become the best deterministic solution due to its intrinsic randomness, but can shift close to the optimum by gradually reducing the exploration rate ϵ

Proof

(2) ϵ-greedy policy

- The update of $\pi^{\epsilon}(a|s)$ satisfies the policy improvement theorem

$$v^{\boldsymbol{\pi^{\epsilon}}}(s) = \sum_{a} \pi^{\epsilon}(a|\mathrm{s}) q^{\boldsymbol{\pi^{\epsilon}}}(s,a)$$

$$= \frac{\epsilon}{|\mathcal{A}|} \sum_{a} q^{\boldsymbol{\pi^{\epsilon}}}(s,a) + (1-\epsilon) \sum_{a} \frac{\pi^{\epsilon}(a|\mathrm{s}) - \epsilon/|\mathcal{A}|}{1-\epsilon} q^{\boldsymbol{\pi^{\epsilon}}}(s,a)$$

Note that a^* gives maximum $q(s,a)$ among all $a \in \mathcal{A}$, thus:

$$\leq \frac{\epsilon}{|\mathcal{A}|} \sum_{a} q^{\boldsymbol{\pi^{\epsilon}}}(s,a) + (1-\epsilon) \max_{a} q^{\boldsymbol{\pi^{\epsilon}}}(s,a)$$

$$= \sum_{a} \boldsymbol{\pi^{\epsilon}_{\mathrm{new}}}(a|\mathrm{s}) q^{\boldsymbol{\pi^{\epsilon}}}(s,a)$$

- Intuitively, ϵ-greedy policy has two parts, of which the random part maintains the same value while the greedy part is improved like greedy policy

Monte Carlo RL

Advantages

- Learn "almost" online from interaction with actual environment
- Only needs complete episodes, and no need to sweep the whole state space
- Usually less harmed by Markovian violations

Disadvantages

- Very slow convergence due to maintain sufficient exploration
- Easy to converge to local optimum due to insufficient exploration

Question: Can model-free MC learn from state-value function?

Yes or No?

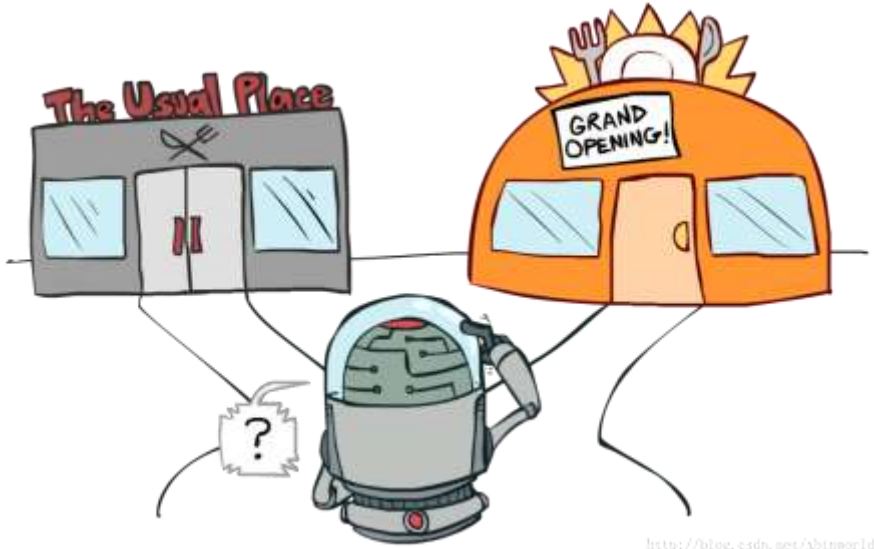
Outline

1. Monte Carlo Estimation
2. MC Algorithm Design
3. On-Policy vs Off-policy
4. Broad Viewpoint of MC

How to Maintain Exploration

Dilemma in model-free RLs

- Want to learn an optimal policy, but need to behave non-optimally in order to explore all potential actions
- **Exploration (探索)** vs **Exploitation (利用)**



Try a new but unknown restaurant? **Go to your favorite restaurant?**

☺ ☺☺ ☺☺☺ or ☺ ☺☺☺

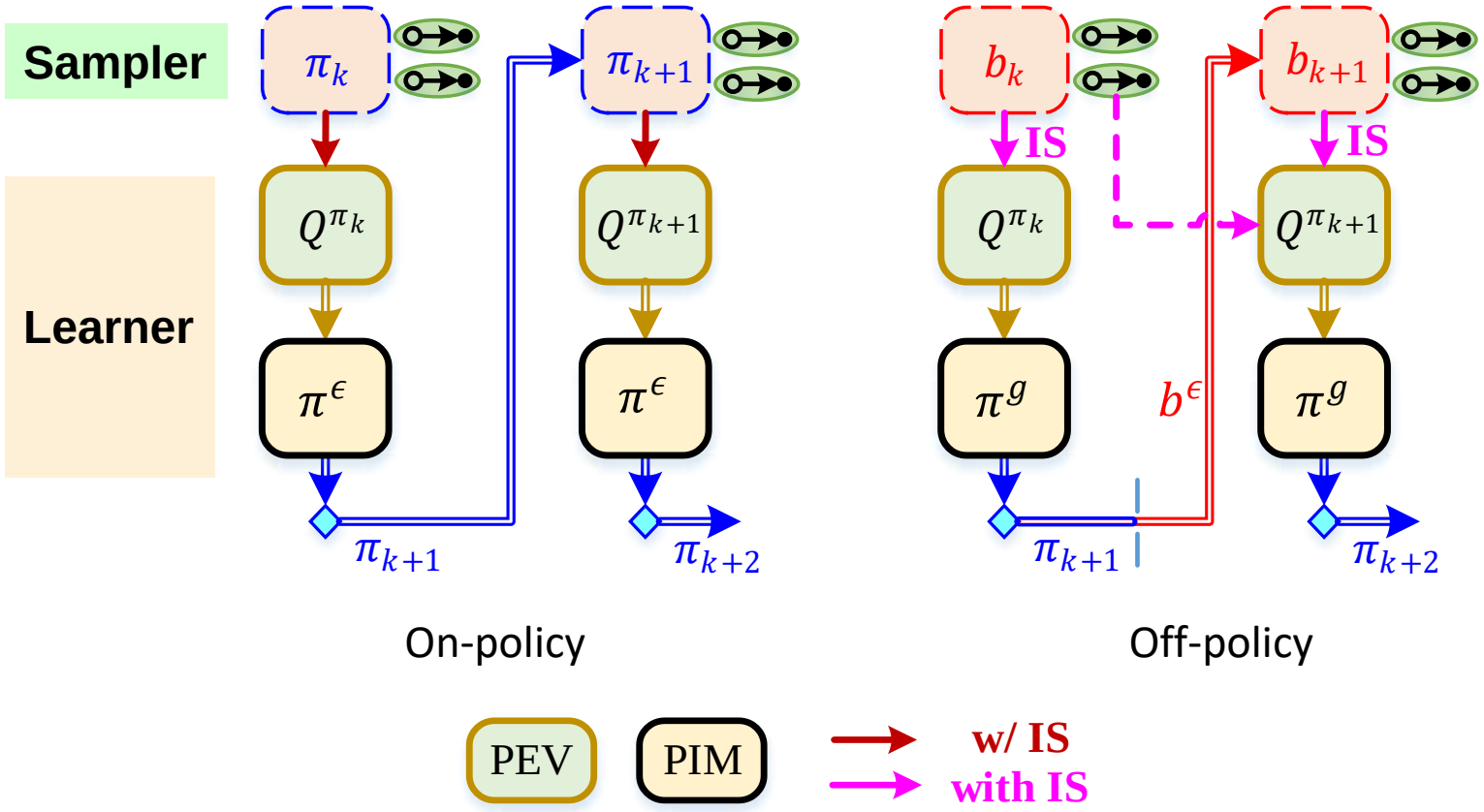
How to Maintain Exploration

- **On-policy strategy**
  - Learn the same policy that is used to explore the environment
  - Must compromise between exploitation and exploration

- **Off-policy strategy**
  - Use different policies in sampler and learner
  - One is greedy to pursue the policy optimality, called **target policy π**
  - The other is soft for environment interaction, called **behavior policy b**

On-policy MC VS Off-policy MC

- **On-policy:** $\pi = b$ vs **Off-policy:** $\pi \neq b$



- Additional merit of off-policy: Allow to use history episodes to estimate current action-value function

Off-policy Design

Off-policy MC learning algorithm

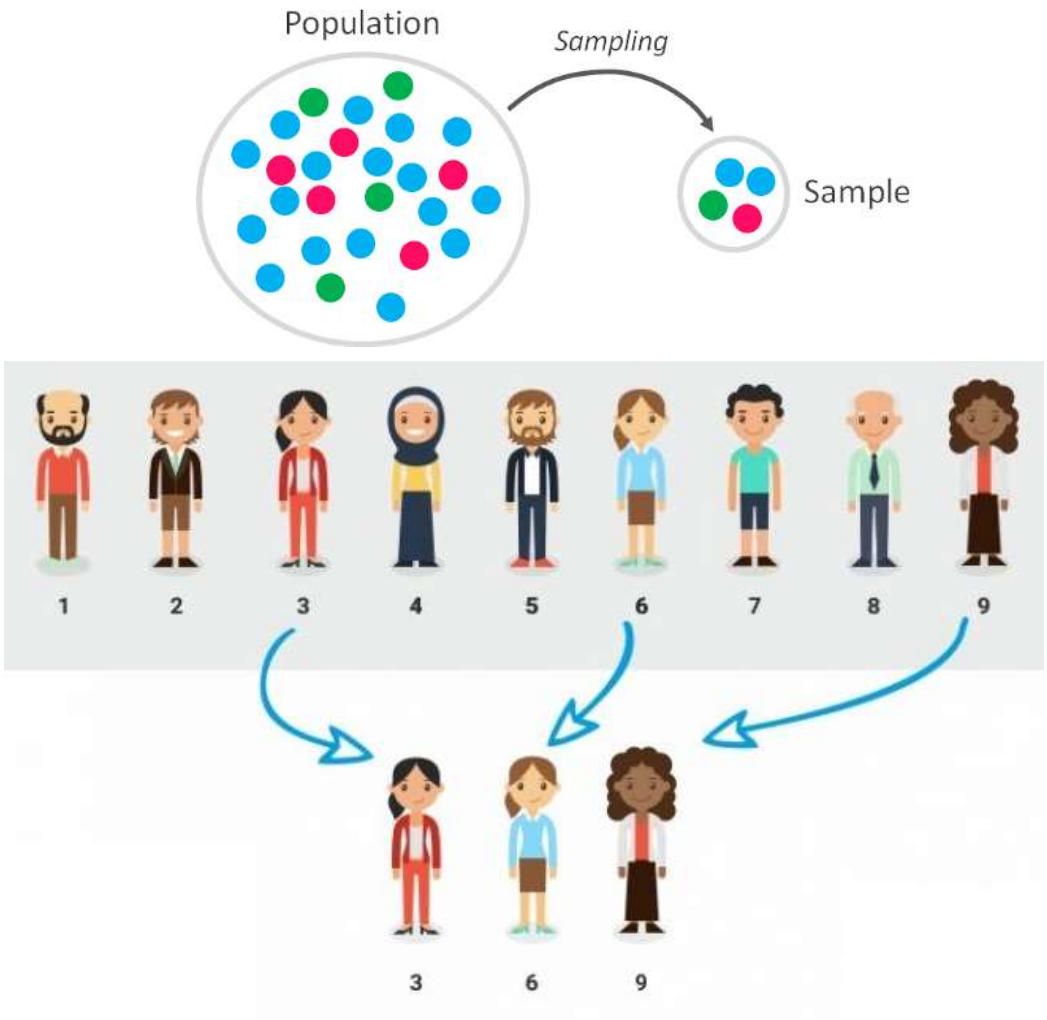
- Target policy π ： to find the best action
- Behavior policy b： to interact with environment

Two issues：

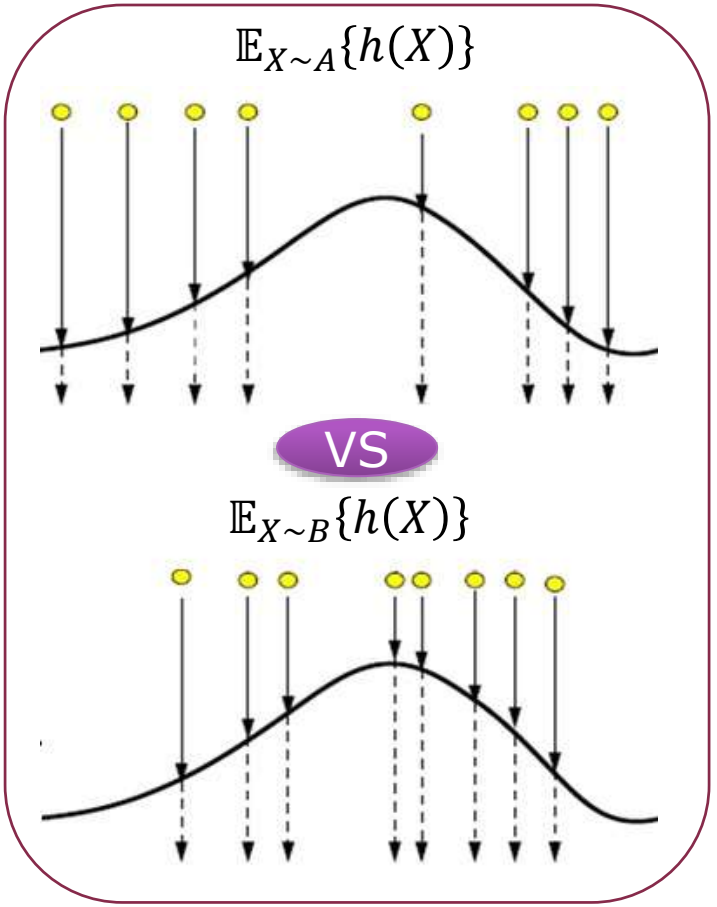
- Assumption of coverage: every action from target policy must be taken at least occasionally under behavior policy
  - π: deterministic or stochastic
  - b: stochastic
- How to bridge the gap between target and behavior policies
  - Utilize importance sampling (IS) technique

Importance Sampling (IS)

- **Sampling really matters in MC estimation**



$\mathbb{E}\{h(X)\}$: Two sets of samples



Importance Sampling (IS)

Principle of Importance Sampling technique

- Achieve identical expectation under different distributions by adding an importance sampling ratio

$$\boldsymbol{\mu_\pi = \mu_b}$$

$\mu_\pi = \mathbb{E}_\pi\{h(x)\}$ and $\mu_b = \mathbb{E}_b\{\rho(x)h(x)\}$

$\rho(x)$ is called importance sampling (IS) ratio

Importance Sampling (IS)

Why IS maintains identical expectations

Select $\rho(x) = \dfrac{d_\pi(x)}{d_b(x)}$

$d_\pi(x)$: target distribution
$d_b(x)$: behavior distribution

$$\mathbb{E}_{x\sim\pi}\{h(x)\} = \int h(x) d_\pi(x) \mathrm{d}x$$
$$= \int h(x) \frac{d_\pi(x)}{d_b(x)} d_b(x) \mathrm{d}x$$
$$= \mathbb{E}_{x\sim b}\{h(x)\rho(x)\}$$

- Corollary: The variance of $h(x)$ under $d_b(x)$ is **NOT** equal to that under $d_\pi(x)$, i.e., $\sigma_b^2 \neq \sigma_\pi^2$

Importance Sampling (IS)

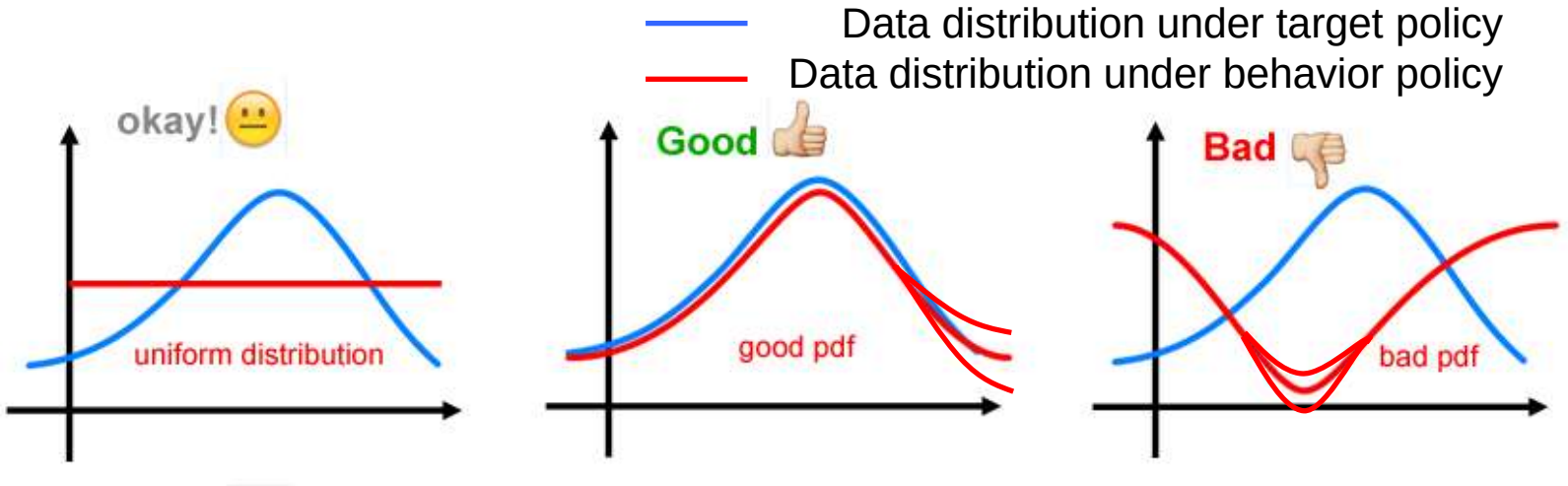
How to utilize IS technique

- Given a trajectory of samples $x_0, x_1, x_2, \cdots, x_{T-1}$ from behavior distribution $d_b(x)$, the estimated expectation under target distribution $d_\pi(x)$ is

$$\hat{\mu}_\pi = \frac{1}{T}\sum_{t=0}^{T-1} h(x_t)\rho(x_t)\,, x_t \sim d_b(x)$$

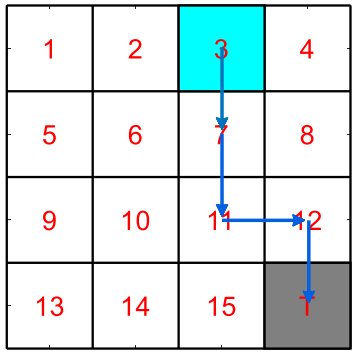
High variance in off-policy estimation

- Large policy mismatch may result in high errors in IS ratio due to the uncertainties in rare behavior actions



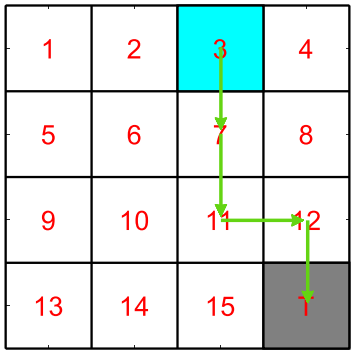
Apply IS to Off-policy MC

- **Same trajectories, but with different probabilities**



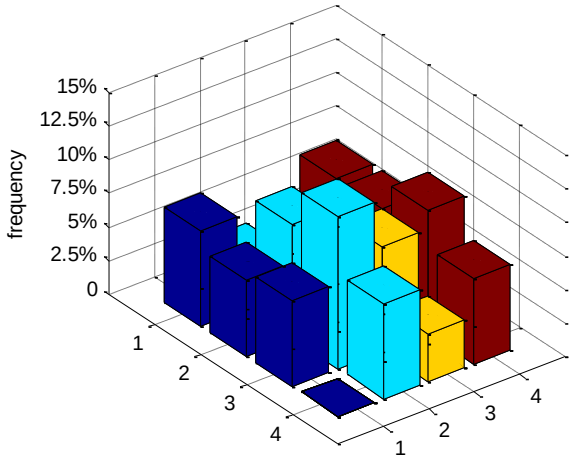
Under policy b

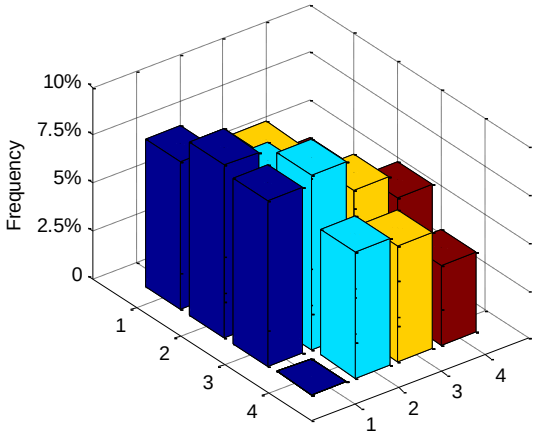
VS



Under policy π

Trajectories have different occurrence probabilities under different policies





Apply IS to Off-policy MC

Off-policy MC

- For a given state s_t, a sample trajectory from t-step to T-step

$$a_t, s_{t+1}, \cdots, a_{T-1}, s_T$$

- The probability mass functions (PMF) under policy π and b are

$$d_\pi(a_t, s_{t+1}, \cdots, a_{T-1}, s_T) =$$
$$\Pr\{s_{t+1}, \cdots, s_T | \boldsymbol{\pi}, s_t\} = \prod_{i=t}^{T-1} \pi(a_i|s_i) p(s_{i+1}|s_i, a_i)$$

$$d_b(a_t, s_{t+1}, \cdots, a_{T-1}, s_T) =$$
$$\Pr\{s_{t+1}, \cdots, s_T | \boldsymbol{b}, s_t\} = \prod_{i=t}^{T-1} b(a_i|s_i) p(s_{i+1}|s_i, a_i)$$

Apply IS to Off-policy MC

Key of off-policy MC

- Estimate value function from target policy distribution given samples from behavior policy distribution
- Weighted returns by **importance sampling (IS) ratio**

$$v^{\pi}(s) = \mathbb{E}_{\pi}\{G_{t:T}|s\} = \mathbb{E}_{b}\{\rho_{t:T-1}G_{t:T}|s\}$$

$$\rho_{t:T-1} = \frac{d_{\pi}(a_t, s_{t+1}, \cdots, a_{T-1}, s_T)}{d_b(a_t, s_{t+1}, \cdots, a_{T-1}, s_T)} = \prod_{i=t}^{T-1} \frac{\pi(a_i|s_i)}{b(a_i|s_i)}$$

- Specially, one-step IS ratio is reduced to

$$\rho_{t:t} = \frac{\pi(a_t|s_t)}{b(a_t|s_t)}$$

Apply IS to Off-policy MC

Proof

- Expectation of $G_{t:T}$ using samples from behavior policy b

$$v^{\pi}(s) = \mathbb{E}_{\pi}\{G_{t:T}|s\}$$

$$= \sum_{(a_t, s_{t+1}, \cdots, a_{T-1}, s_T)} d_{\pi}(a_t, s_{t+1}, \cdots, a_{T-1}, s_T)\, G_{t:T}(a_t, s_{t+1}, \cdots, a_{T-1}, s_T)\Big|_{s_t=s}$$

$$= \sum_{(a_t, s_{t+1}, \cdots, a_{T-1}, s_T)} \frac{d_{\pi}(*)}{d_b(*)} d_b(*)\, G_{t:T}(*)\Big|_{s_t=s}$$

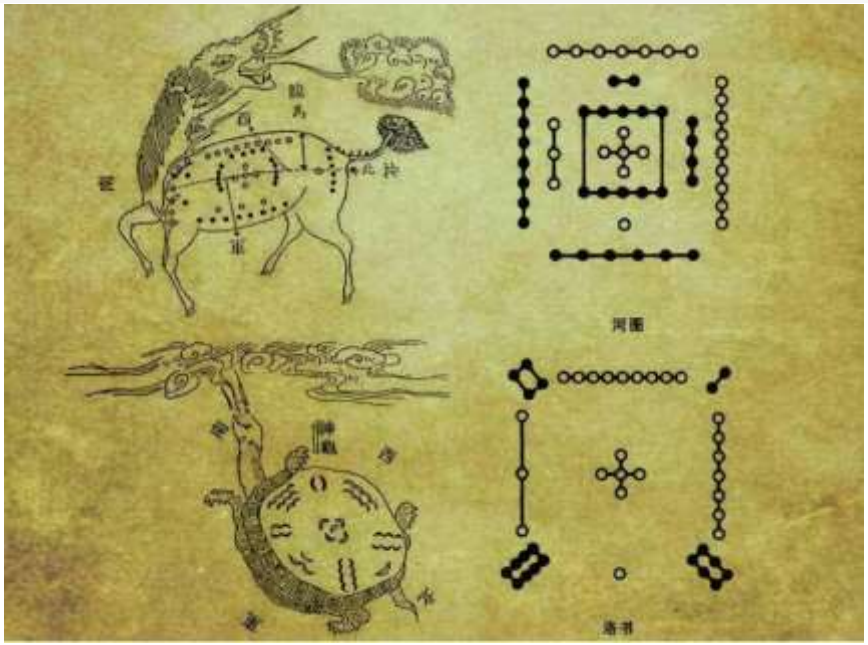
$$= \sum_{(a_t, s_{t+1}, \cdots, a_{T-1}, s_T)} d_b(*)\rho_{t:T-1}\, G_{t:T}(*)\Big|_{s_t=s}$$

$$= E_{b}\{\rho_{t:T-1} G_{t:T}|s\}$$

Outline

1. Monte Carlo Estimation
2. MC Algorithm Design
3. On-Policy vs Off-policy
4. Broad Viewpoint of MC

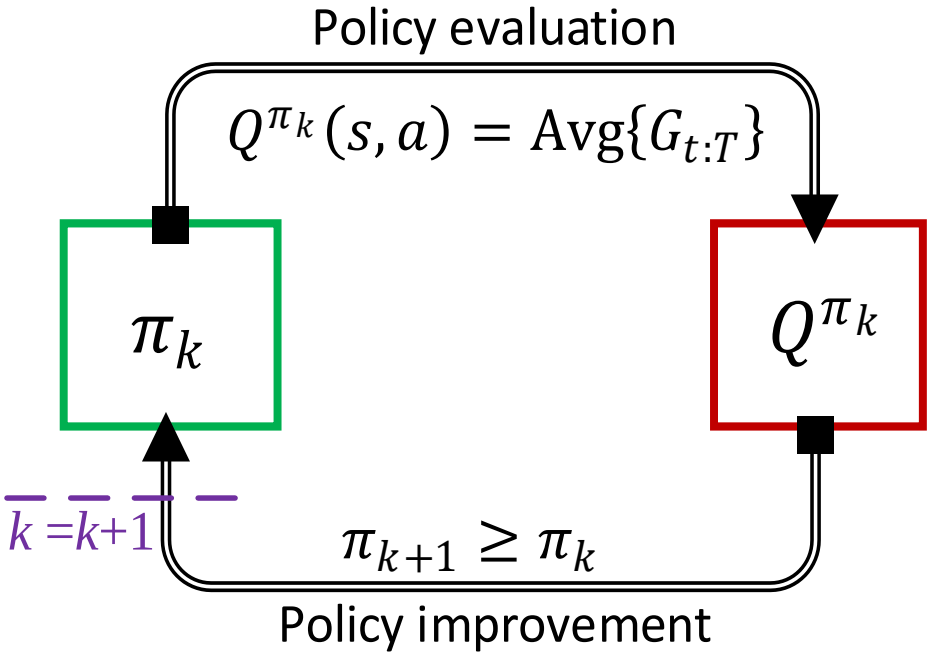
Yin-Yang diagram



Fuxi & Nuwa (Tang dynasty in China)

Cycle in Monte Carlo RL

Alternating Cycle of PEV and PIM



$$\pi_0 \xrightarrow{\text{PEV}} Q^{\pi_0} \xrightarrow{\text{PIM}} \pi_1 \xrightarrow{\text{PEV}} Q^{\pi_1} \xrightarrow{\text{PIM}} \cdots \xrightarrow{\text{PIM}} \pi^* \xrightarrow{\text{PEV}} q^*$$

On-policy MC

On-policy batch estimation

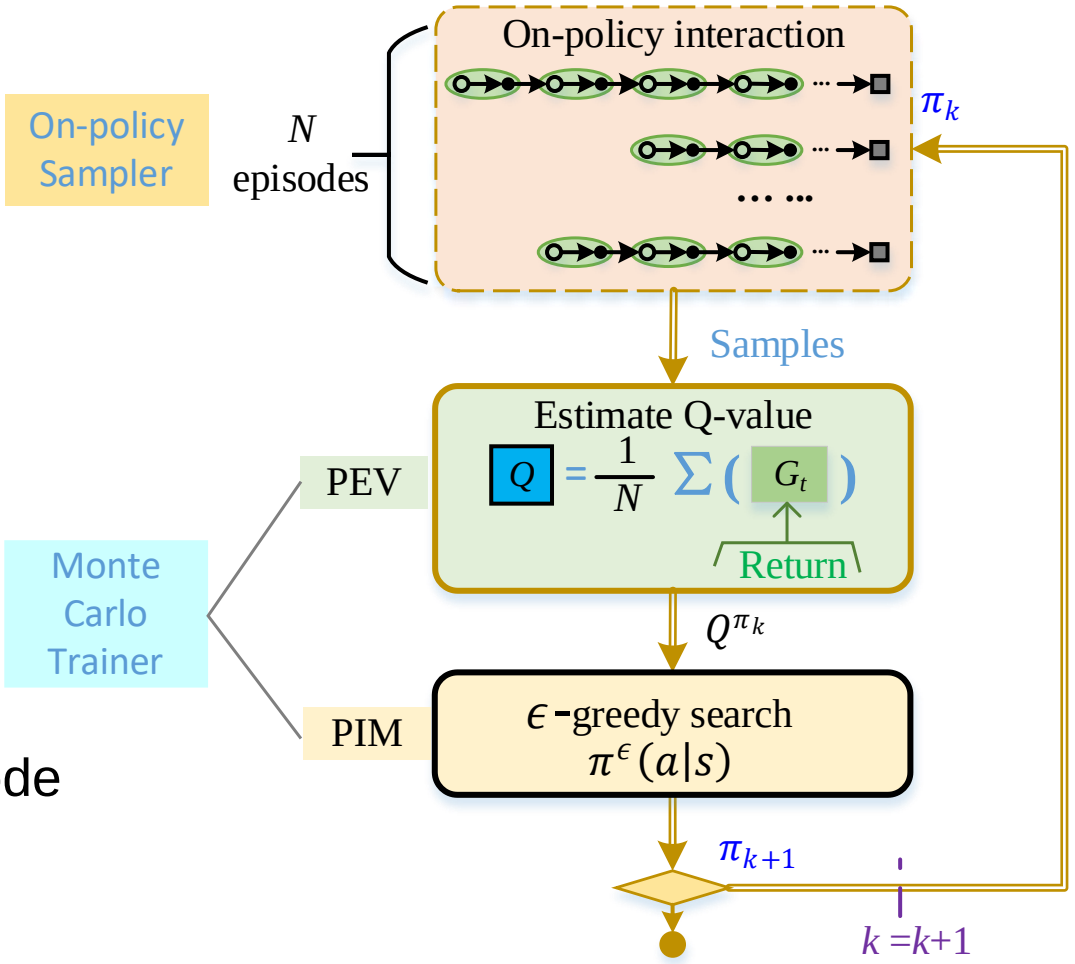
- Generates a series of episodes (i.e., batch episodes)

$$G^{(1)}_{t_1:T}, G^{(2)}_{t_2:T}, \cdots, G^{(i)}_{t_i:T}, \cdots G^{(N)}_{t_N:T}$$

- Batch average at each state-action pair (s, a)

$$Q^{\pi}(s,a) = \frac{1}{N}\sum_{i=1}^{N} G^{(i)}_{t_i:T}$$
$$\forall s_{t_i} = s, a_{t_i} = a$$

- N > 1: batch estimation
- N = 1: episode-by-episode estimation



Off-policy MC

Off-policy batch estimation

- Generates a series of episodes with behavior policies

- Returns and IS ratios

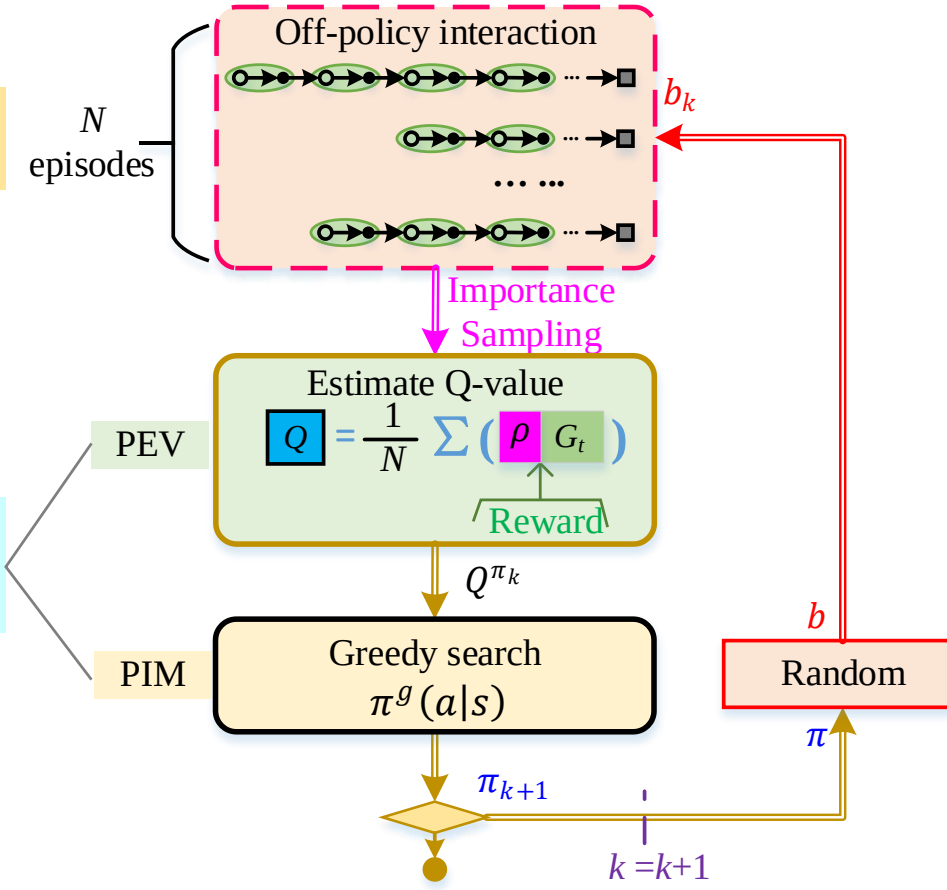
$$G_{t_1:T}^{(1)}, G_{t_2:T}^{(2)}, \cdots, G_{t_i:T}^{(i)}, \cdots G_{t_N:T}^{(N)}$$

$$\rho_{t_1+1:T-1}^{(1)}, \cdots, \rho_{t_i+1:T-1}^{(i)}, \cdots \rho_{t_N+1:T-1}^{(N)}$$

- Ordinary batch average

$$Q^{\pi}(s,a) = \frac{1}{N}\sum_{i=1}^{N} \rho_{t_i+1:T-1}^{(i)} G_{t_i:T}^{(i)}$$

$$\forall s_{t_i} = s, a_{t_i} = a$$

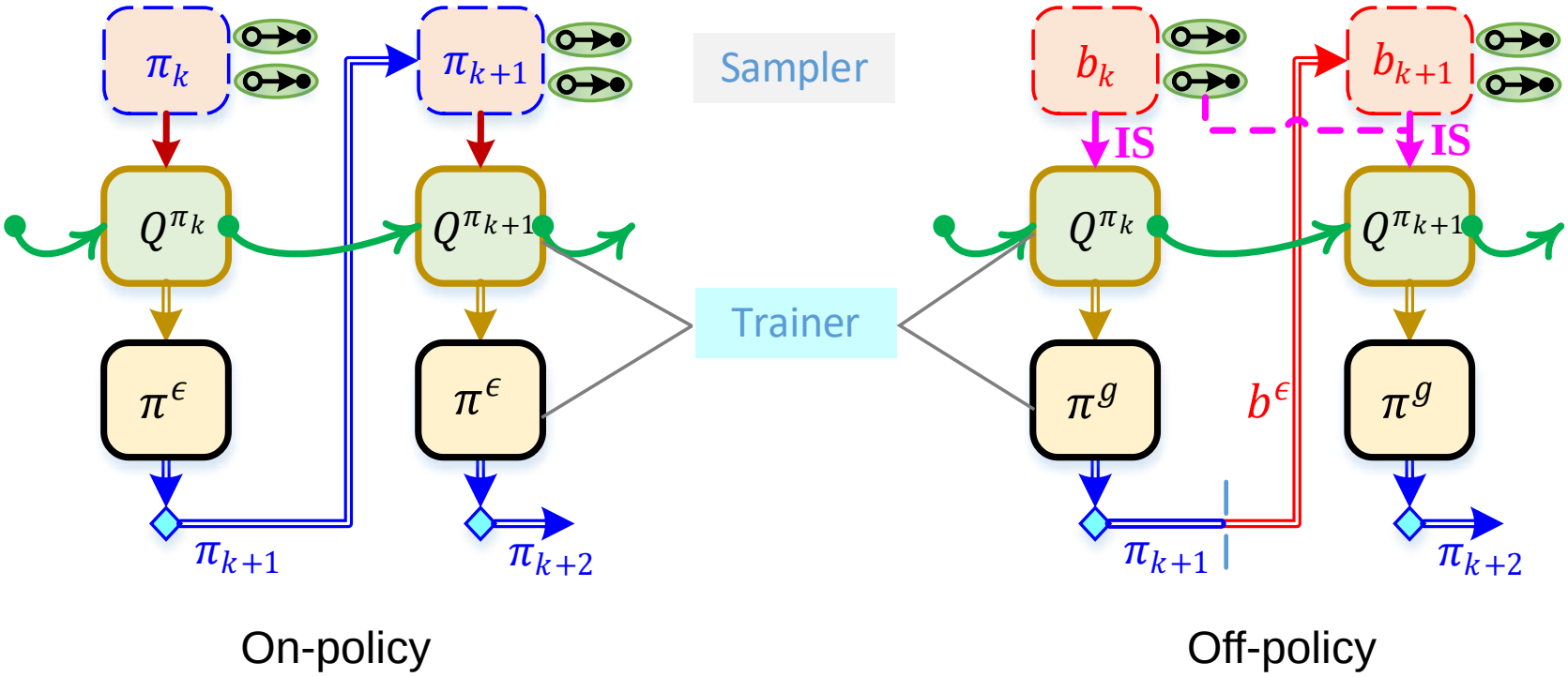


Value function initialization

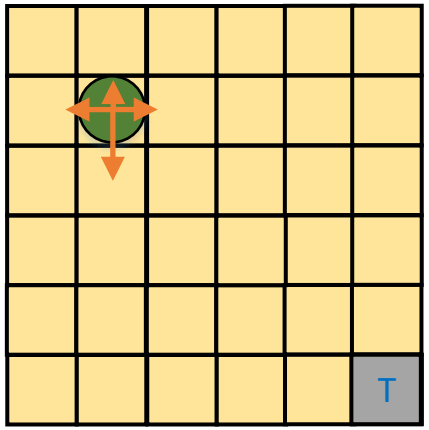
Incremental MC estimation

$$Q^{\pi_{k+1}}(s,a) = (1-\lambda)Q^{\pi_k}(s,a) + \lambda \cdot \mathrm{Avg}\{G|s,a;\pi_{k+1}\},$$

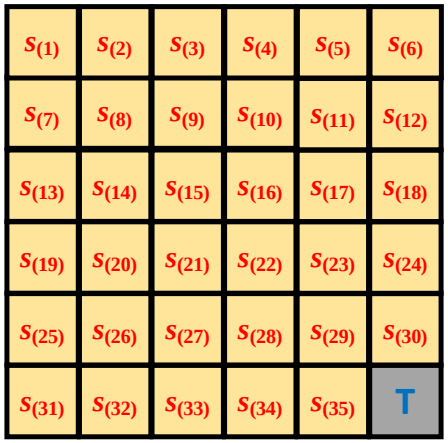
$\lambda \in (0,1]$ is called incremental rate



Example: cleaning robot

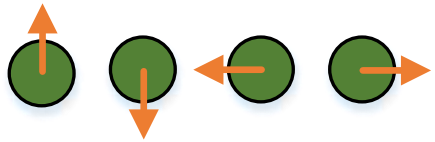


Grid environment



State = $\{s_{(i)}\}$, i=1, 2,… , 36

State space

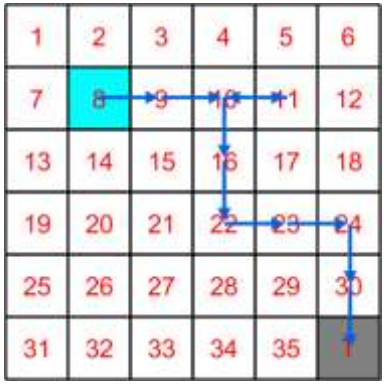


Action = {North, South, West, East}

Action space

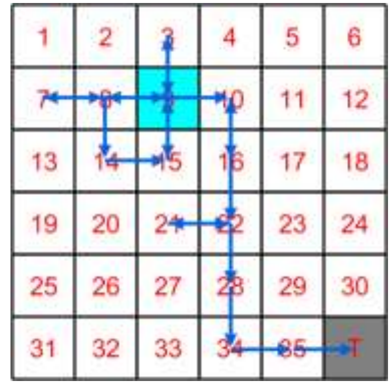
- On-policy agent with ϵ-greedy policy, where $\epsilon = 0.05$
- Episodes used in each PEV, i.e., Ep/PEV = 50
- Discount factor = 0.95

Example: cleaning robot



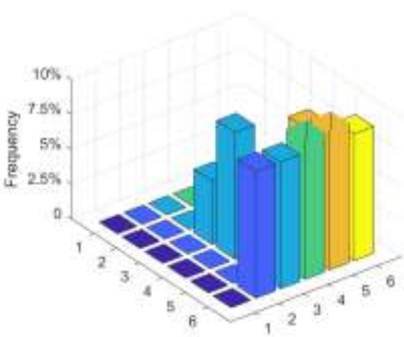
episode

episode

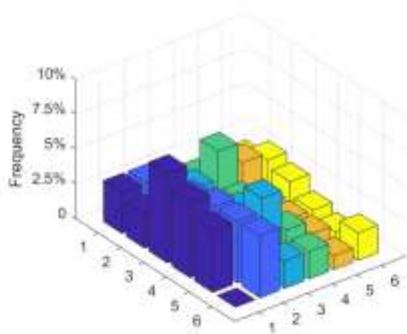


episode

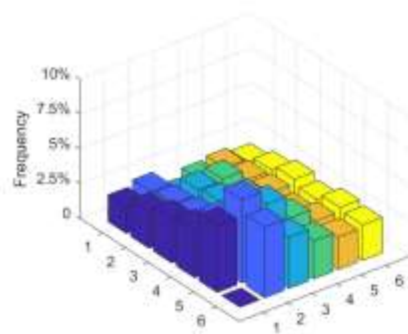
episode



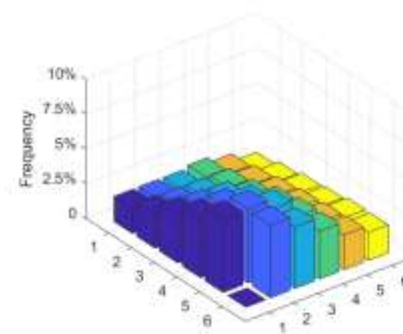
after 1 episode



after 10 episodes



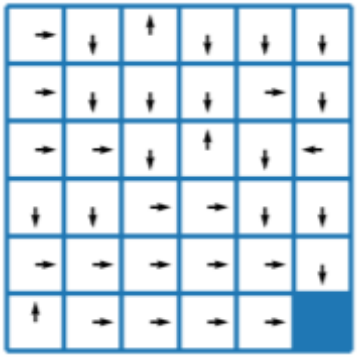
after 20 episodes



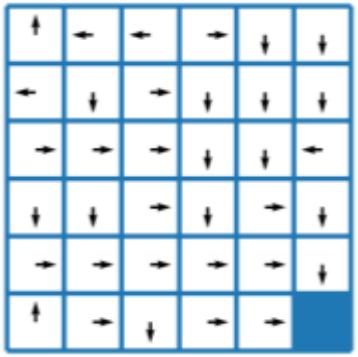
after 50 episodes

Example: cleaning robot

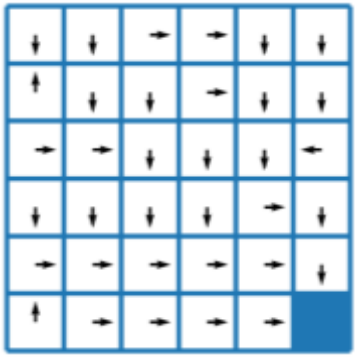
Intermediate policy



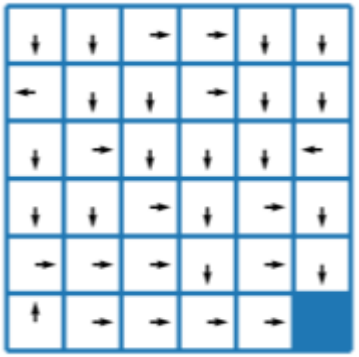
k=1



k=2



k=3



k=4

Estimated state-value function

-3.4	-2.6	-2.4	-2.4	-2.8	-2.2
-3.5	-2.8	-2.0	-1.7	-1.5	-1.4
-3.0	-2.3	-1.9	-1.6	-1.2	-1.2
-3.1	-2.4	-1.9	-1.3	-0.6	0.2
-2.9	-2.2	-1.4	-0.2	0.9	1.6
-2.4	-1.6	-1.3	0.2	1.6	

k=1

-3.4	-3.1	-2.5	-2.7	-2.9	-2.4
-3.6	-3.0	-2.6	-2.4	-1.2	-1.3
-3.3	-2.2	-1.5	-0.8	-0.5	-0.8
-2.7	-1.9	-1.4	0.2	0.6	1.5
-1.9	-1.0	0.1	1.2	2.3	3.3
-1.4	-0.4	0.1	1.7	3.3	

k=2

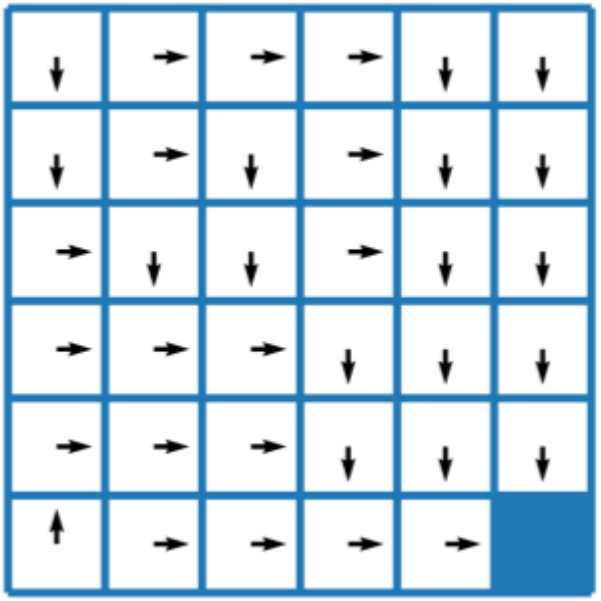
-4.8	-4.0	-1.9	-1.9	-2.2	-2.2
-4.0	-3.0	-2.0	-1.7	-0.7	-1.7
-3.1	-2.2	-0.8	-0.3	0.2	-0.1
-2.6	-1.2	-0.7	1.0	1.5	2.7
-1.9	-0.8	0.6	2.2	3.4	4.6
-1.7	-1.4	0.7	2.9	4.7	

k=3

-5.5	-4.5	-2.7	-2.1	-1.8	-2.2
-4.3	-3.1	-1.9	-1.8	-0.4	-1.3
-3.4	-1.9	-0.5	-0.1	0.7	0.3
-3.2	-1.1	0.2	1.3	2.3	3.6
-2.3	-0.6	1.0	2.9	4.0	5.6
-2.4	-0.3	1.7	3.7	5.6	

k=4

Example: cleaning robot



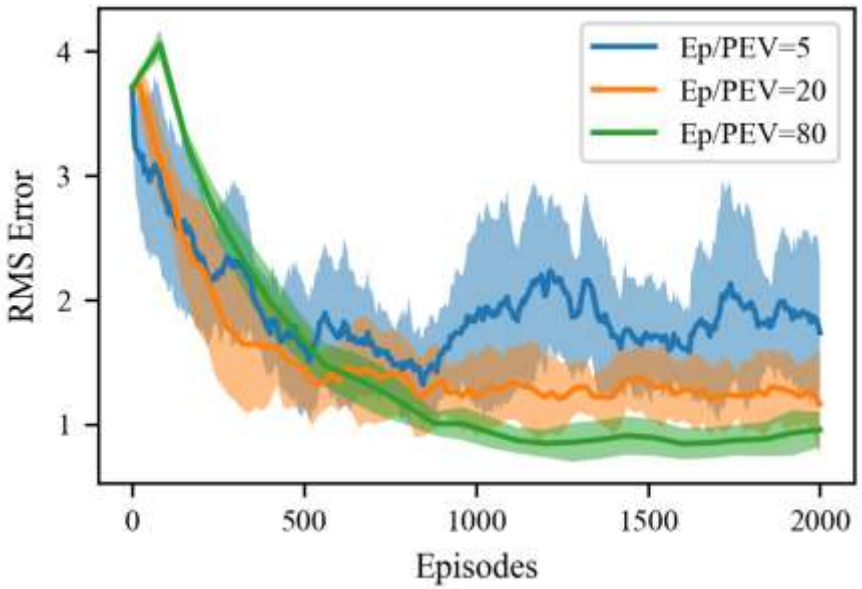
Final policy

-5.6	-3.6	-2.4	-1.6	-0.4	0.9
-3.8	-3.3	-2.1	-0.2	1.2	2.5
-3.5	-0.9	-0.0	0.8	3.4	4.5
-1.9	-0.0	1.3	2.9	4.6	6.3
-0.4	1.2	2.3	4.5	6.6	8.5
-0.7	3.1	4.3	6.0	8.4	

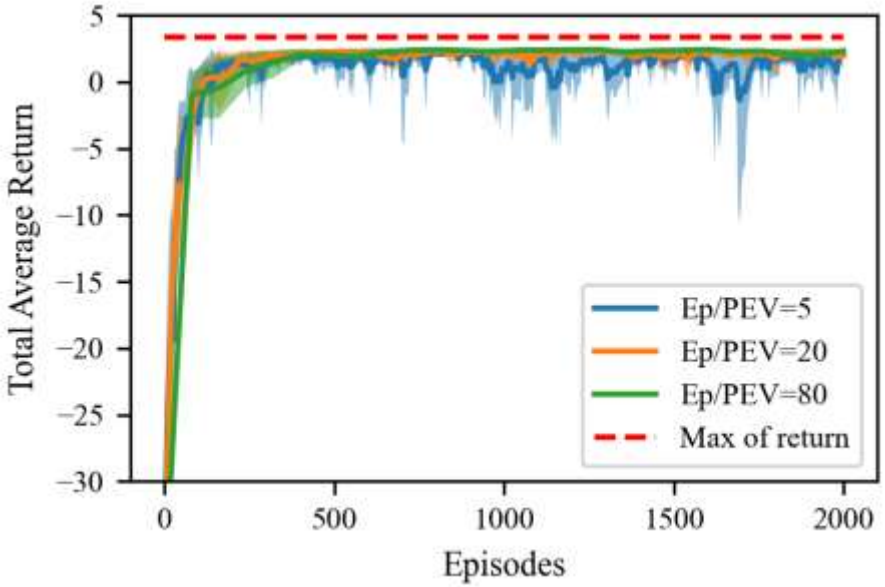
Final value function

Example: cleaning robot

Influence of episode number per PEV



(a) RMS error

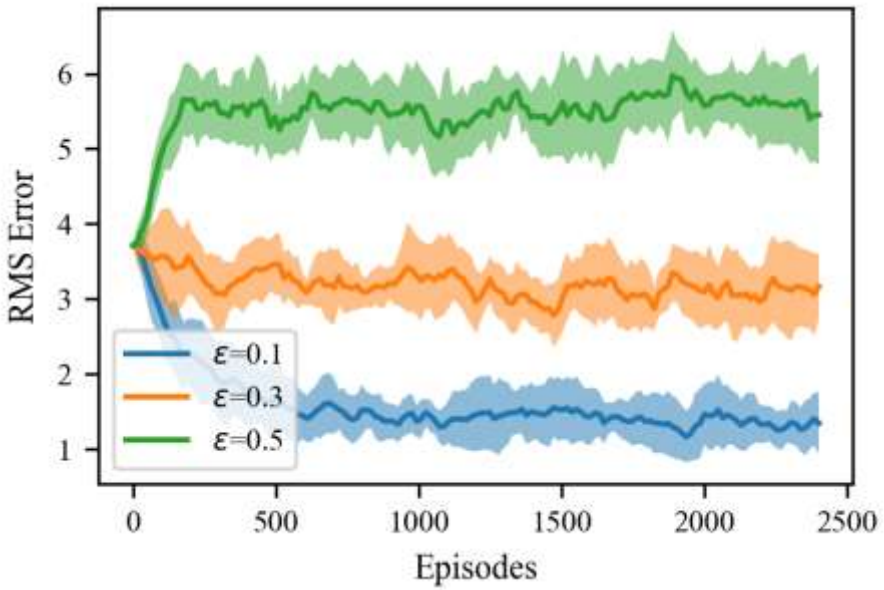


(b) Total average return

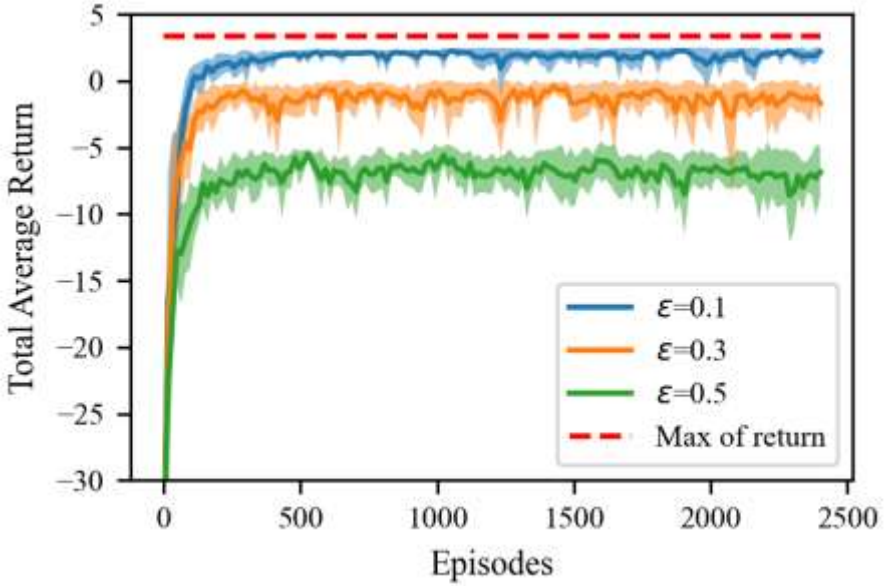
- A large Ep/PEV can reduce the RMS error and improve the learning stability, and does not change the convergence rate very much even though the frequency of policy improvement becomes sparse in the same number of total episodes

Example: cleaning robot

Influence of exploration rate



(a) RMS error

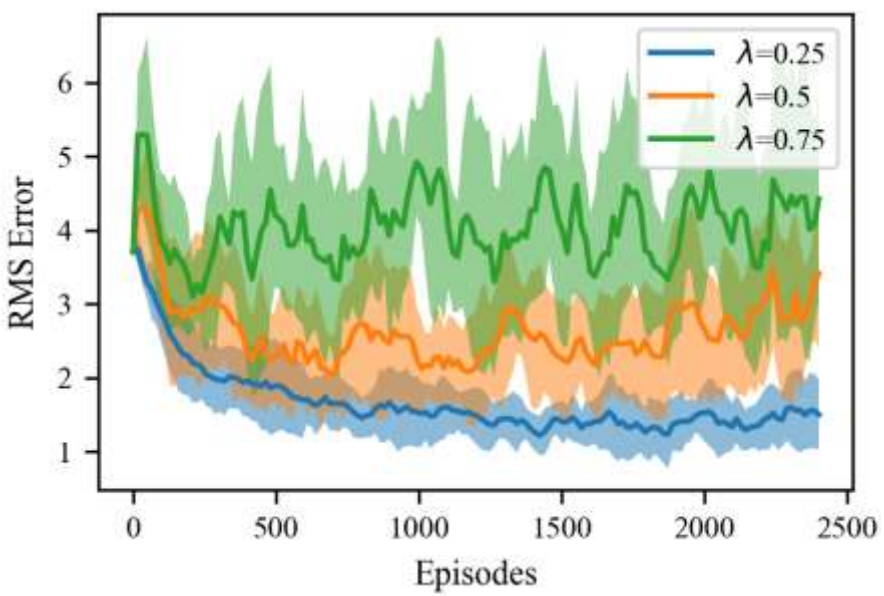


(b) Total average return

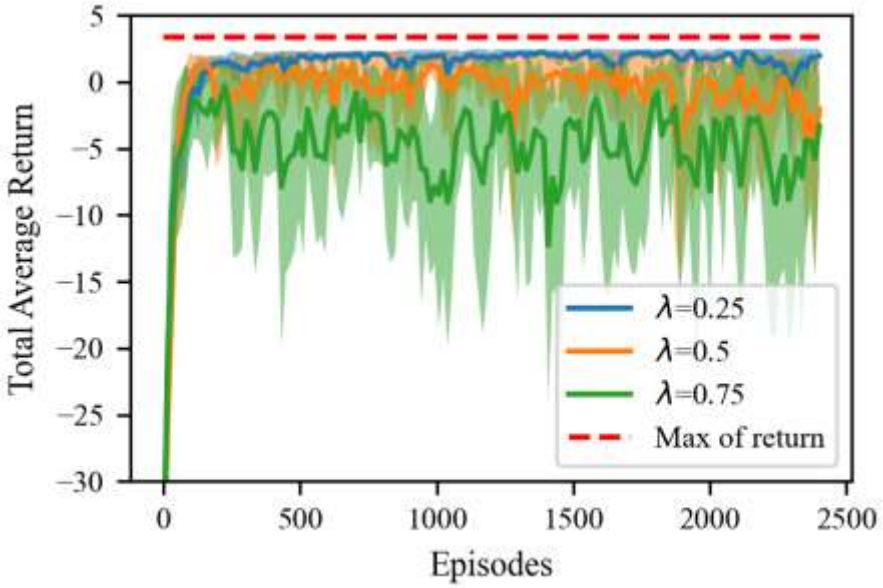
- The exploration rate reflects the ability to balance exploration and exploitation
- A large exploration rate has a negative impact on the policy optimality because the optimal policy should be deterministic but a too large exploration rate will make almost a random policy

Example: cleaning robot

Influence of incremental rate



(a) RMS error



(b) Total average return

- The incremental rate is a key parameter to determine the effectiveness of value initialization
- A small incremental rate implies more reliance on the historical experience, which is helpful in stabilizing the training process and improving the policy performance

The End!