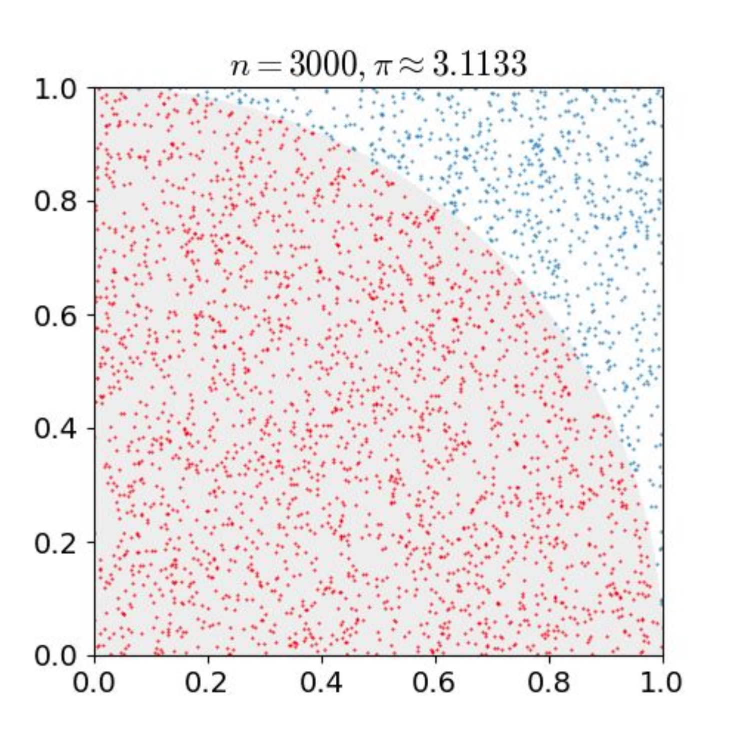
Chapter 06. 스스로 전략을 짜는 강화학습 (Reinforcement Learning)

Monte Carlo Methods



Bellman Equation

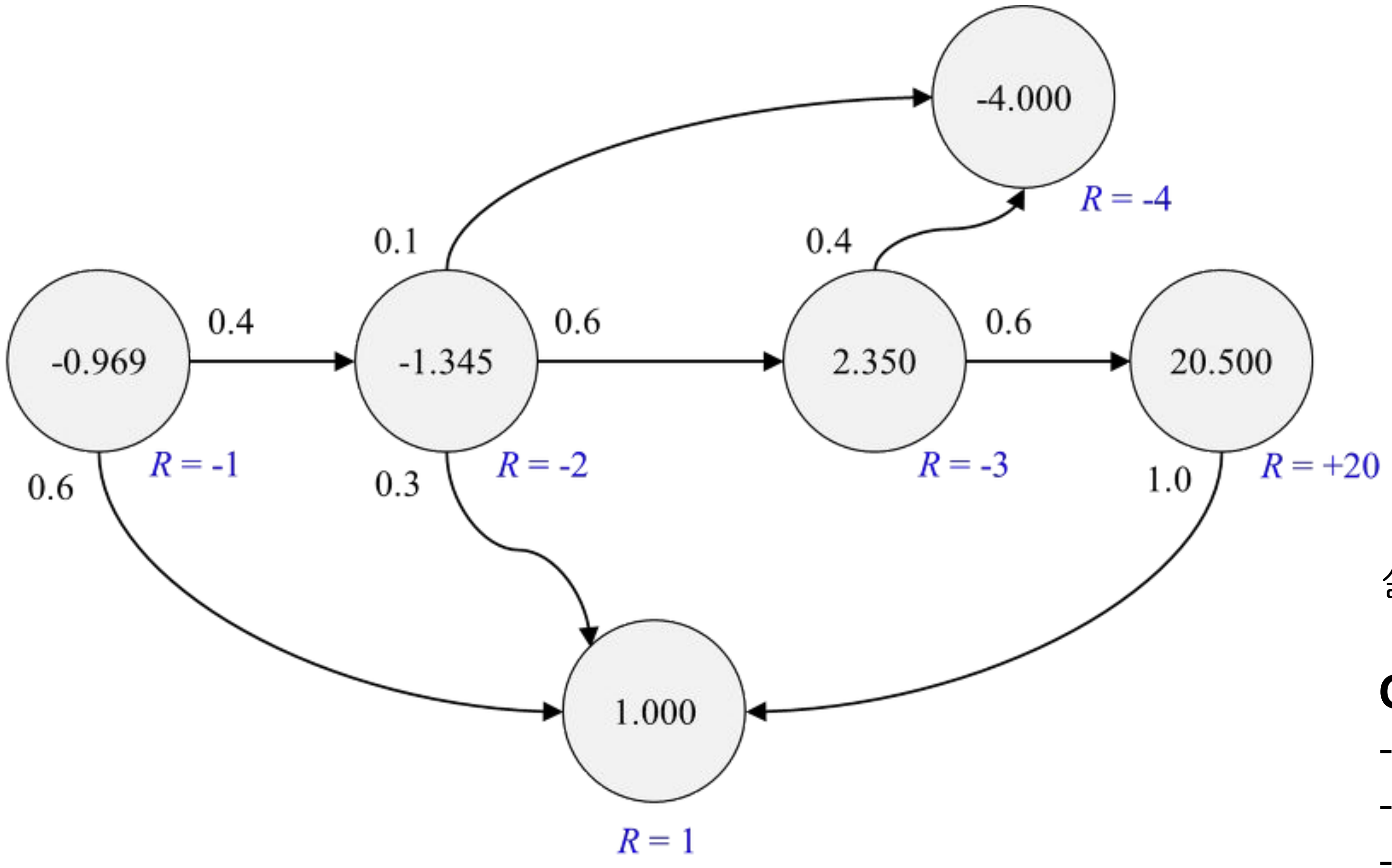
The value function can be decomposed into two parts:

- immediate reward R_{t+1}
- discounted value of successor state $\gamma v(S_{t+1})$

$$
\begin{aligned}
v(s) &= \mathbb{E}\left[G_t \mid S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma\left(R_{t+2} + \gamma R_{t+3} + \dots\right) \mid S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s\right]
\end{aligned}
$$

Markov Reward Process

https://untitledtblog.tistory.com/139



실행 시간 복잡도 : **O (n³)**

Other iterative solving method
-**Dynamic Programming**
-**Temporal Difference Learning**
-**Monte**-**Carlo Method**

Markov Decision Process

State-value function with policy

MDP에서 state-value function v(s)는 Markov reward process의 state-value function과 마찬가지로 state s에서 시작했을 때 얻을 수 있는 return의 기댓값을 의미한다. 그러나 MDP는 주어진 policy π를 따라 action을 결정하고, state를 이동하기 때문에 MDP에서의 state-value function은 다음과 같이 정의된다

$$v_\pi(s) = E_\pi[G_t|S_t=s] = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t=s]$$

$$= \sum_{a\in A} \pi(a|s)\left(r(s,a) + \gamma \sum_{s'\in S} p(s'|s,a)v_\pi(s')\right) \qquad (11)$$

Dynamic Programming

Problem	Bellman Equation	Algorithm
Prediction	Bellman Expectation Equation	Iterative Policy Evaluation
Control	Bellman Expectation Equation + Greedy Policy Improvement	Policy Iteration
Control	Bellman Optimality Equation	Value Iteration

- Algorithms are based on state-value function $v_\pi(s)$ or $v_*(s)$
- Complexity $O(mn^2)$ per iteration, for m actions and n states
- Could also apply to action-value function $q_\pi(s,a)$ or $q_*(s,a)$
- Complexity $O(m^2n^2)$ per iteration

Planning & Learning

•Planning

앞서 배운 environment에 대한 model 을 가지고 있는 경우, Markov Decision Process 에 대한 full knowlege 를 가지고 있게 된다. 이를 planning 이라고 하며 MDP 의 정보를 기반한다.

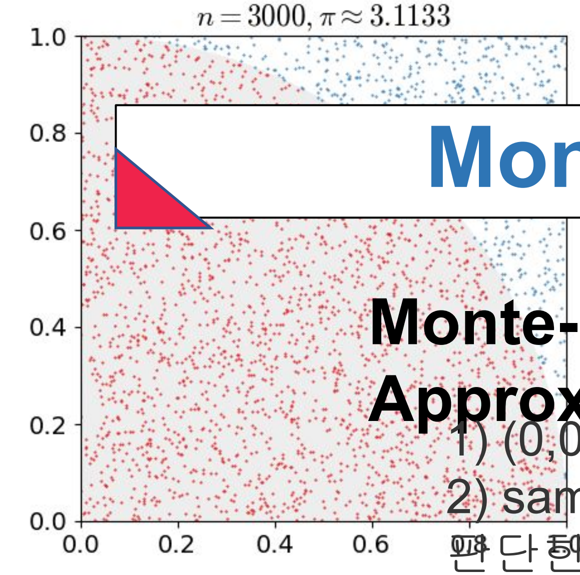
•Learning

Learning이란 environment의 model을 모르지만 상호작용을 통해서 문제를 푸는 경우를 말합니다.

•Process of Planning

- For prediction:
  - Input: MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ and policy π
  - or: MRP $\langle \mathcal{S}, \mathcal{P}^{\pi}, \mathcal{R}^{\pi}, \gamma \rangle$
  - Output: value function v_{π}
- Or for control:
  - Input: MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$
  - Output: optimal value function v_{*}
  - and: optimal policy π_{*}

fast campus



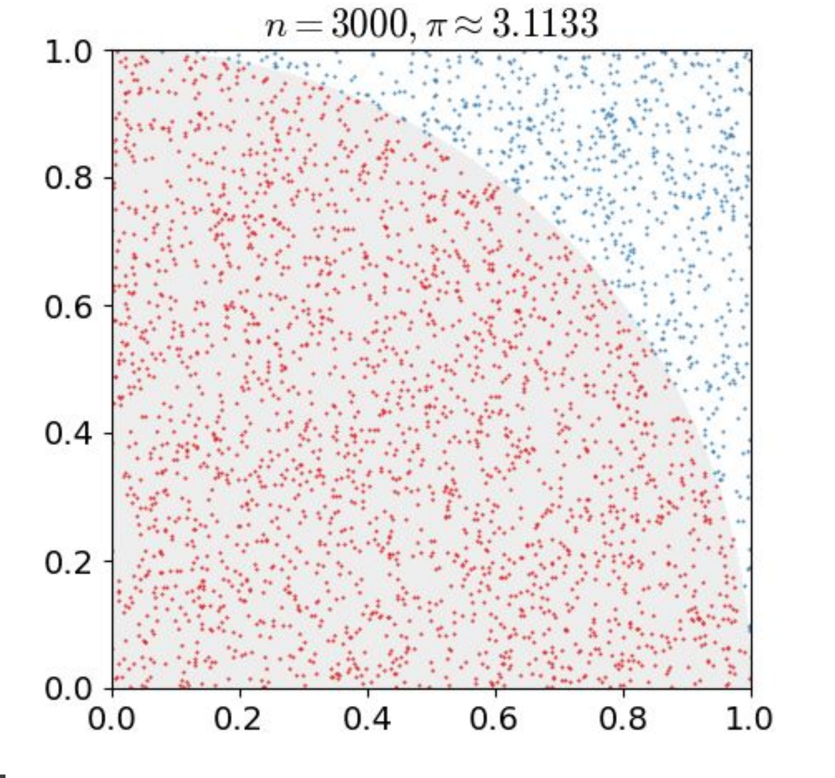
Monte Carlo Method

https://sumniya.tistory.com/11?category=781573

Monte-Carlo Approximation

1) (0,0), (0,1), (1,0), (1,1) 로 이루어진 좌표 상에 임의의 점 (x, y)를 sampling한다.
2) sampling한 점이 중심이 (0,0)이고 반지름이 1인 사분원 내에 속하는 점인지를 판단한다.
3) 이 과정을 충분히 반복한다.

$$n \rightarrow \infty \, , \quad \frac{1}{n}\sum_{i=1}^{n} I(dot_i \in quadrant) \sim \frac{\pi}{4}$$



Monte Carlo Method

Episode

$$S_1 \xrightarrow{R_1} S_2 \xrightarrow{R_2} S_2 \xrightarrow{R_3} \cdots \longrightarrow S_T$$

$$\mathrm{G}(s_1) = R_1 + \gamma R_2 + \gamma^2 R_3 + \cdots$$

$$\mathrm{G}(s_2) = R_2 + \gamma^2 R_3 + \cdots$$

$$\mathrm{G}(s_3) = R_3 + \cdots$$

Dynamic Programming

$$v(s) = \mathbb{E}\left[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s\right]$$

Monte Carlo Method

$$V_\pi(s) \sim \frac{1}{N(s)} \sum_{i=1}^{N(s)} G_i(s)$$

Monte Carlo Method

$$V_{n+1} = \frac{1}{n}\sum_{i=1}^{n} G_i = \frac{1}{n}\left(G_n + \sum_{i=1}^{n-1} G_i\right)$$

$$= \frac{1}{n}\left(G_n + \sum_{i=1}^{n-1} G_i\right) = \frac{1}{n}\left(G_n + (n-1)\frac{1}{(n-1)}\sum_{i=1}^{n-1} G_i\right)$$

$$= \frac{1}{n}(G_n + (n-1)V_n)$$

$$= \frac{1}{n}(G_n + nV_n - V_n)$$

$$= V_n + \frac{1}{n}(G_n - V_n)$$

Incremental method

step-si
ze

erro
r

$$V(s) \leftarrow V(s) + \frac{1}{n}(G(s) - V(s)) \qquad V(s) \leftarrow V(s) + \boxed{\alpha}\,\boxed{(G(s) - V(s))}$$

Monte Carlo Method

First Visit Method

- To evaluate state s
- The first time-step t that state s is visited in an episode,
- Increment counter $N(s) \leftarrow N(s) + 1$
- Increment total return $S(s) \leftarrow S(s) + G_t$
- Value is estimated by mean return $V(s) = S(s)/N(s)$
- By law of large numbers, $V(s) \rightarrow v_\pi(s)$ as $N(s) \rightarrow \infty$

Monte Carlo Method

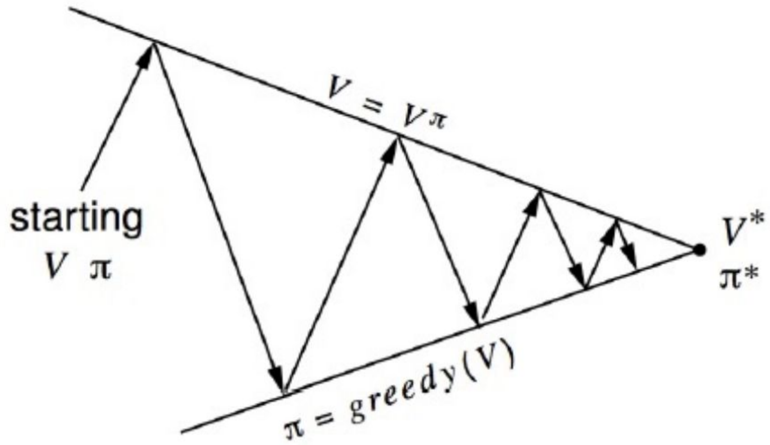
Every Visit Method

- To evaluate state s
- Every time-step t that state s is visited in an episode,
- Increment counter $N(s) \leftarrow N(s) + 1$
- Increment total return $S(s) \leftarrow S(s) + G_t$
- Value is estimated by mean return $V(s) = S(s)/N(s)$
- Again, $V(s) \rightarrow v_\pi(s)$ as $N(s) \rightarrow \infty$

Monte Carlo Method

Monte-Carlo Control

Policy Iteration *= (policy evaluation + policy improvement)*
Monte-Carlo policy Iteration *= (MC policy evaluation + policy improvement)*



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

Problem 1 . Value Function -> MDP

$$v(s) = \mathcal{R}_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'} v(s')$$

Problem 2 . Greedy policy improvement
-> Local Optimum

Monte Carlo Method

Problem 1 . Value Function -> MDP

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

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

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

$$\pi'(s) = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ Q(s,a)$$

Problem 2 . Greedy policy improvement -> Local Optimum

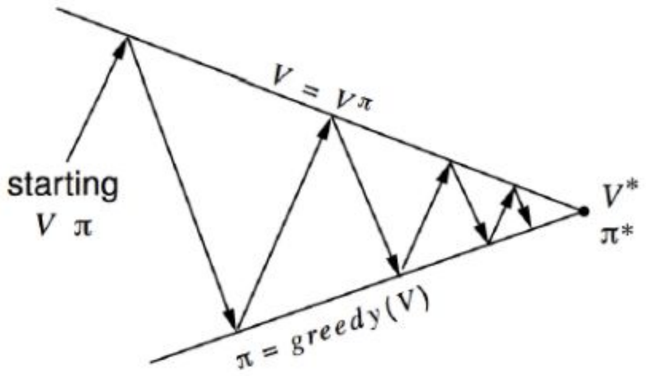
$$\pi(s) \doteq \arg\max_a q(s,a).$$

$$\begin{aligned} q_{\pi_k}(s, \pi_{k+1}(s)) &= q_{\pi_k}(s, \arg\max_a q_{\pi_k}(s,a)) \\ &= \max_a q_{\pi_k}(s,a) \\ &\geq q_{\pi_k}(s, \pi_k(s)) \\ &\geq v_{\pi_k}(s). \end{aligned}$$

- 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

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

Monte Carlo Method

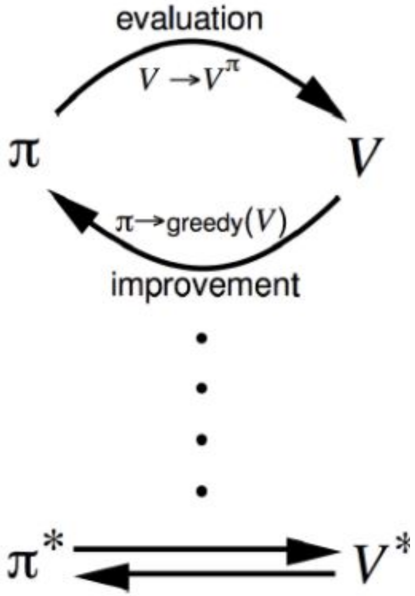


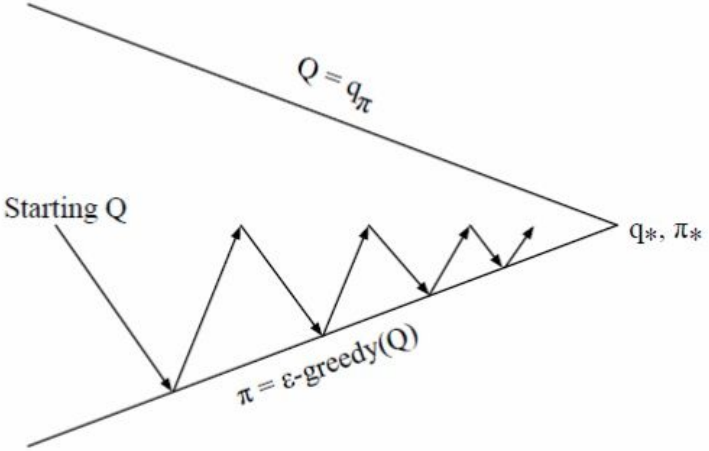
Policy evaluation Estimate v_π

e.g. Iterative policy evaluation

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

e.g. Greedy policy improvement





Every episode:

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

Policy improvement ϵ-greedy policy improvement

Monte Carlo Method

Monte Carlo ES (Exploring Starts), for estimating $\pi \approx \pi_*$

Initialize:
- $\pi(s) \in \mathcal{A}(s)$ (arbitrarily), for all $s \in \mathcal{S}$
- $Q(s, a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$
- $Returns(s, a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Loop forever (for each episode):
- Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}(S_0)$ randomly such that all pairs have probability > 0
- Generate an episode from S_0, A_0, following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
- $G \leftarrow 0$
- Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
  - $G \leftarrow \gamma G + R_{t+1}$
  - Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:
    - Append G to $Returns(S_t, A_t)$
    - $Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$
    - $\pi(S_t) \leftarrow \text{argmax}_a \, Q(S_t, a)$

fast campus

- *Thank you*