Deep Reinforcement Learning

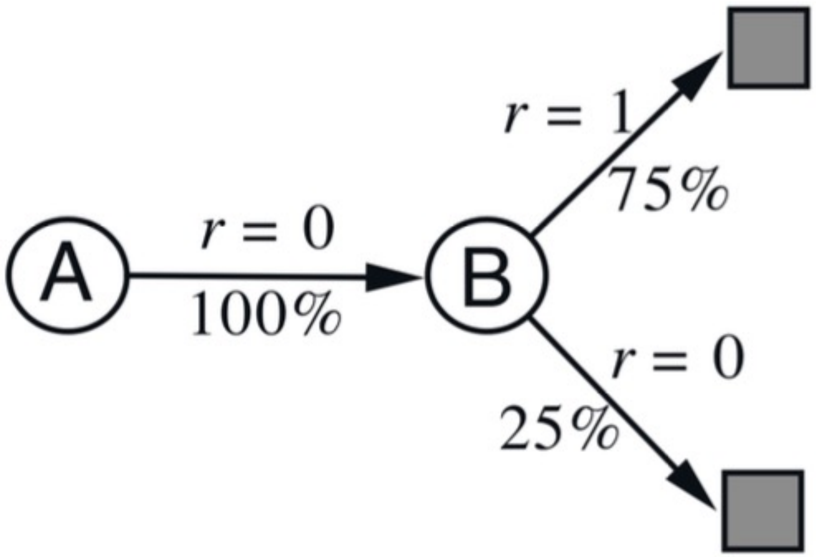
14, 15 Temporal Difference learning

Yun Seong Cho

목차

AB example

AB example



AB example for unknown MDP with a batch of 8 episodes.

$$\{A, 0, B, 0\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 0\}$$

AB example

$$\{A, 0, B, 0\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 0\}$$

What is the best values for the estimates $V(A)$ and $V(B)$? $(\gamma\ = 1)$

AB example

$\{A, 0, B, 0\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 0\}$

What is the best values for the estimates $V(A)$ and $V(B)$? $(\gamma = 1)$

MC $V(A) \leftarrow V(A) + \alpha\,[\underline{R_{t+1} + \gamma V(B)} - V(A)]$

TD $V(A) \leftarrow V(A) + \alpha\,[\underline{G_t} - V(A)]$

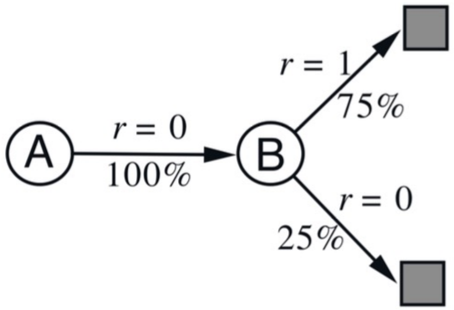
AB example

$$\{A, 0, B, 0\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 0\}$$

What is the best values for the estimates $V(B)$? ($\gamma = 1$)

AB example

$\{A, 0, B, 0\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 0\}$



What is the best values for the estimates $V(B)$? $(\gamma = 1)$

MC 시작점(S0)가 B이면 에피소드가 끝날 때까지 진행하면 expected return = 0.75

TD 시작점(S0)가 B이면 한번 액션을 취하고 얻는 expected reward + γ x estimate = 0.75

⟶ **MC, TD : V(B) = 0.75**

AB example

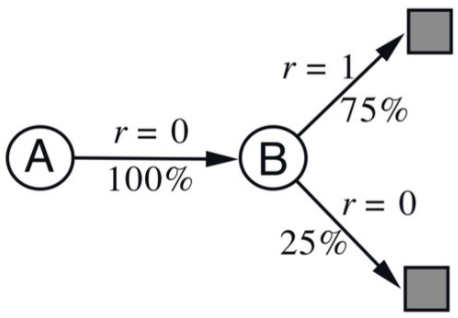
$\{A, 0, B, 0\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 0\}$

What is the best values for the estimates $V(A)$ $(\gamma = 1)$

AB example

$\{A, 0, B, 0\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 1\}$, $\{B, 0\}$

What is the best values for the estimates $V(A)$ $(\gamma = 1)$



MC 시작점(S0)가 A이면 에피소드가 끝날 때까지 진행하면 expected return = 0

TD 시작점(S0)가 A이면 한번 액션을 취하고 얻는 expected reward + γ x estimate = 0.75

→ **MC : V(A) = 0 , TD : V(A) = 0.75**

AB example

$$\{A, 0, B, 0\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 1\},\ \{B, 0\}$$

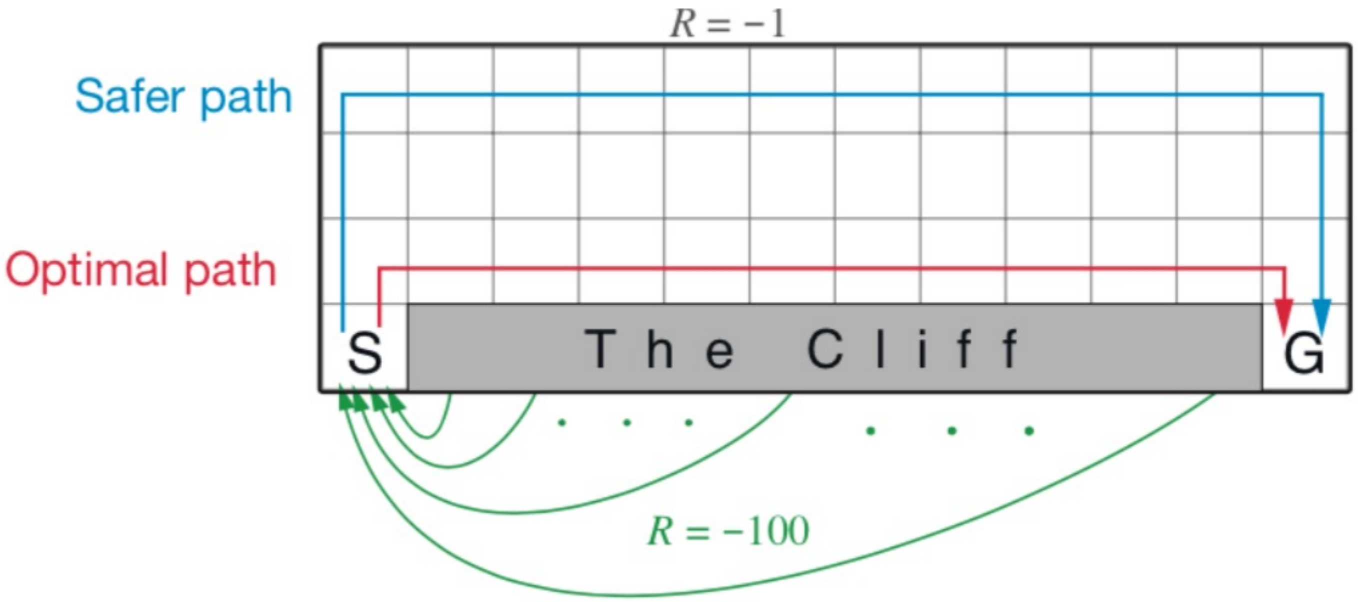
What is the best values for the estimates $V(A)$ $(\gamma = 1)$

MC	V(A) = 0	V(B) = 0.75
TD	V(A) = 0.75	V(B) = 0.75

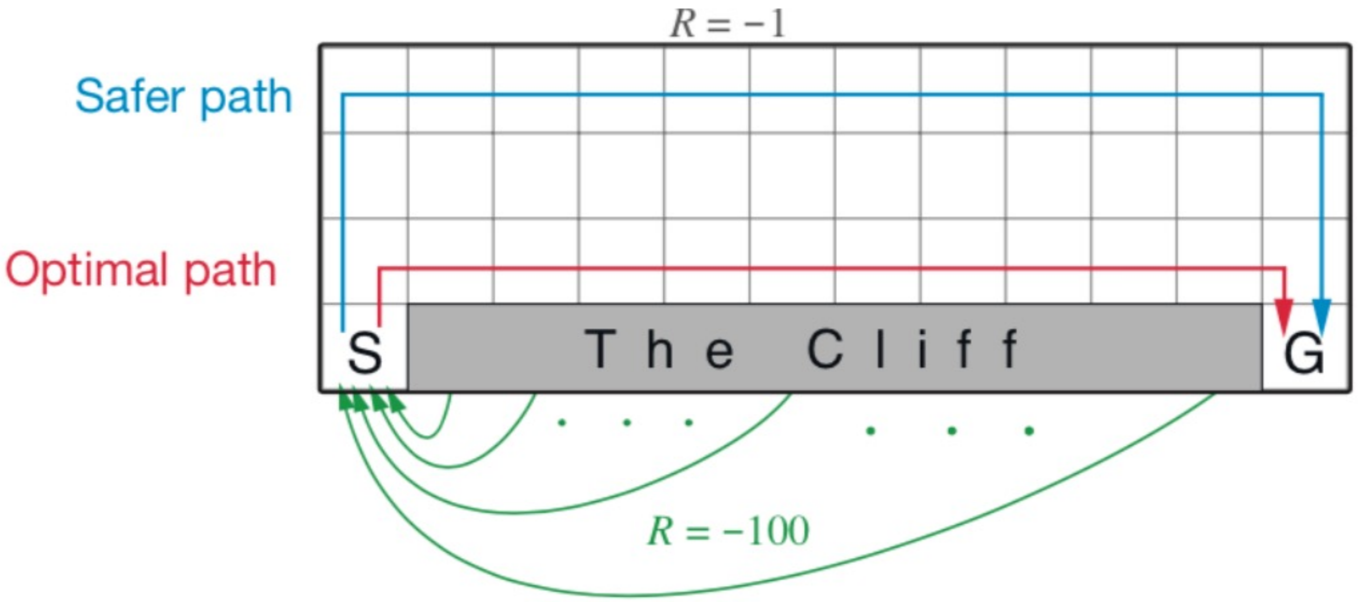
→ MC 방법이 sample에 더 의존하고, TD 방법은 estimate을 사용하여 sample의 future data error에 저항이 있다.

Cliff walking

Cliff walking



Cliff walking



Sarsa vs Q-learning

Cliff walking

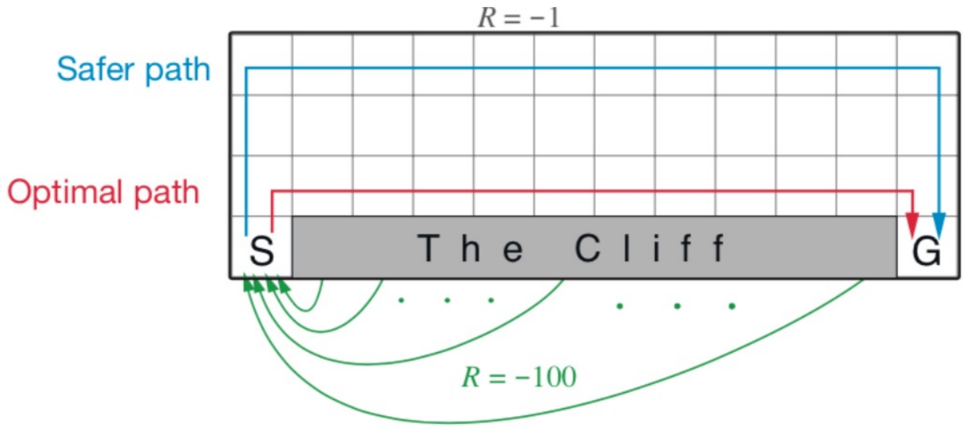
Sarsa: On-Policy TD Prediction

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

Q-learning: Off-Policy TD Prediction (Watkins 1989)

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

$$\varepsilon - greedy(Q) = \begin{cases} 1 - \varepsilon + \dfrac{\varepsilon}{m} & , a : highest\ Q \\ \dfrac{\varepsilon}{m} & , otherwise \end{cases}$$



Cliff walking

Sarsa: On-Policy TD Prediction

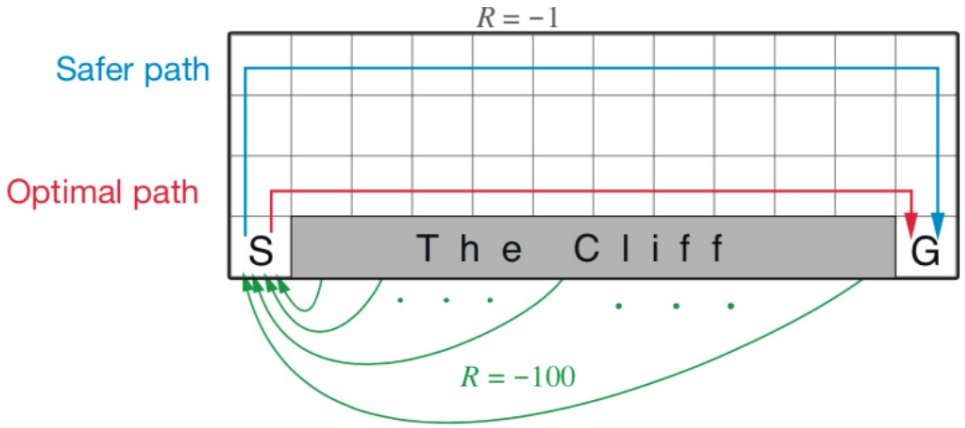
$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t) \right]$$

' target policy가 $\varepsilon - greedy(Q)$ 방식을 사용하기 때문에 절벽 아래로 떨어질 확률을 고려 '

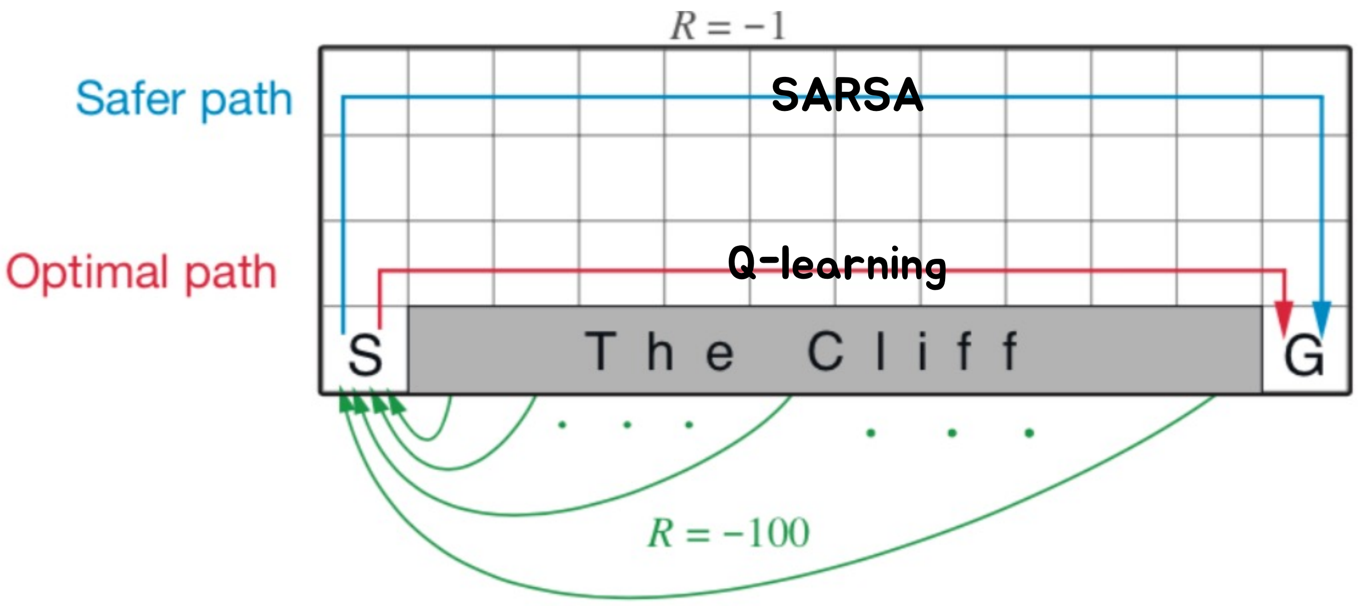
Q-learning: Off-Policy TD Prediction (Watkins 1989)

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

'target policy가 $\max_a Q(S_{t+1}, a)$ 방식을 사용하기 때문에 이익이 최대가 되는 방향만 고려 '

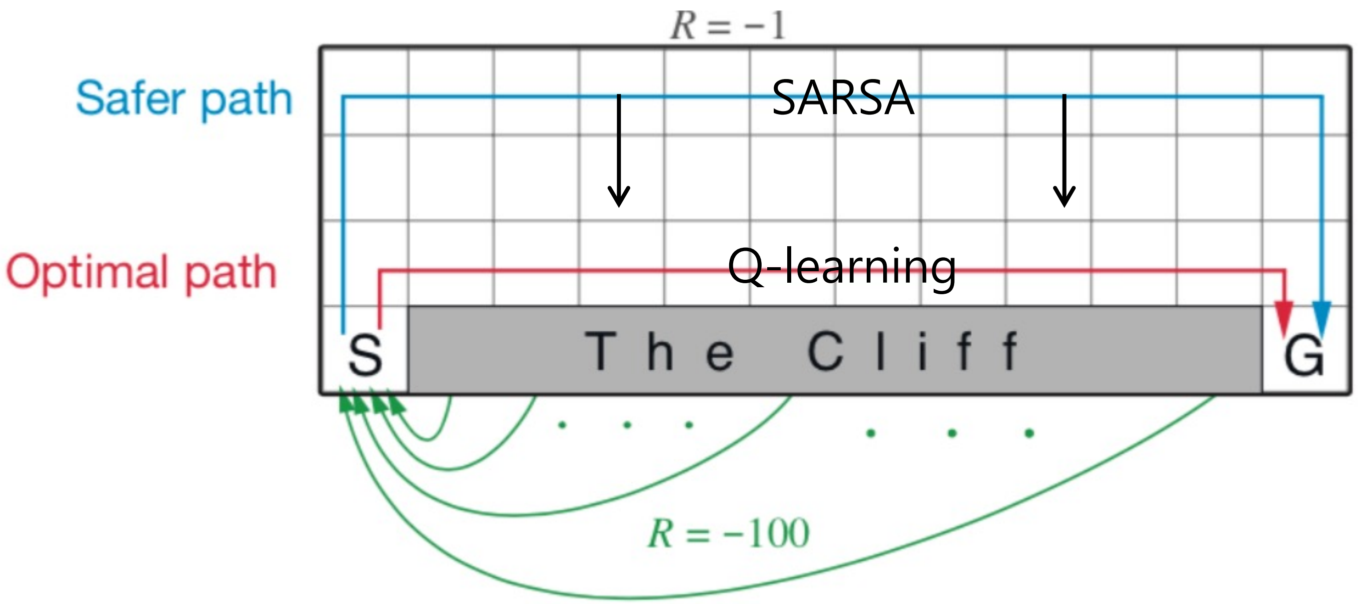


Cliff walking

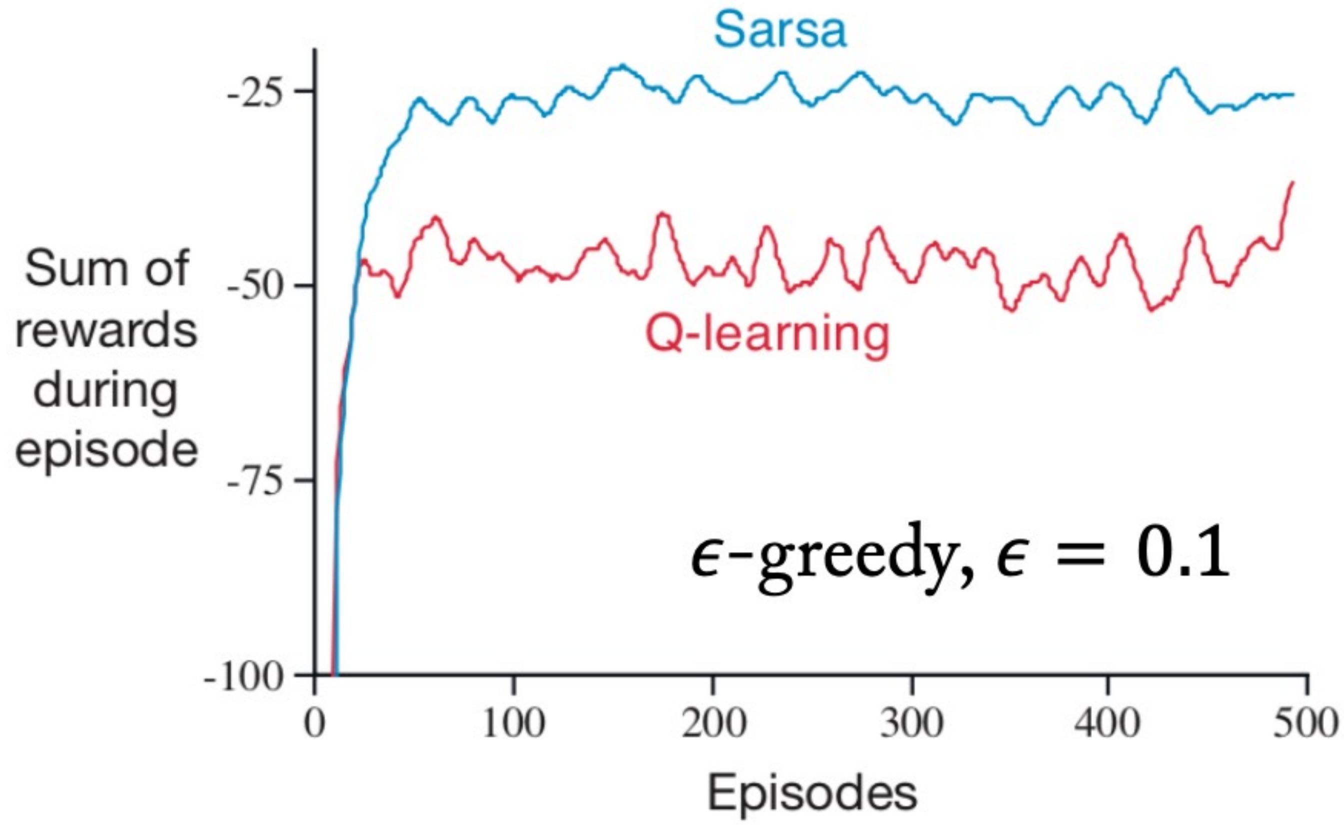


Cliff walking

만약 ε이 0에 가까워진다면, SARSA에서 다른 선택을 고려하지 않게 되고, 그로 인해 optimal path로 수렴



Cliff walking



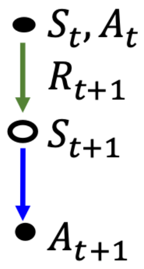
Variation of method

Expected Sarsa

$$\varepsilon - greedy(Q) = \begin{cases} 1 - \varepsilon + \dfrac{\varepsilon}{m} & , a : highest\ Q \\ \dfrac{\varepsilon}{m} & , otherwise \end{cases}$$

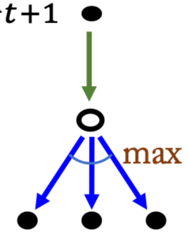
- Sarsa

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



- Q-learning

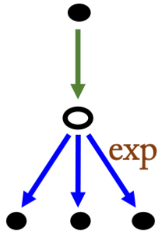
$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right]$$



- Expected Sarsa

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \mathbb{E}_\pi [Q(S_{t+1}, A_{t+1}) \mid S_{t+1}] - Q(S_t, A_t) \right]$$

$$\left(= Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \sum \pi(a \mid S_{t+1}) Q(S_{t+1}, a) - Q(S_t, A_t) \right] \right)$$



기존 Sarsa는 비록 $\varepsilon - greedy(Q)$ 방식을 사용하였지만, 그 중에 하나의 액션에 대한 값을 target으로 정하게 된다.

이것을 보완하기 위하여 같은 policy에 대해서 expected sum을 한 값을 target으로 정해서 모든 액션을 고려할 수 있다.

Double Q-learning

기존 Q-learning은 PE 단계에서 maximization operation을 사용하기 때문에 실제 sampling 할 때와 비교했을 때 과대평가(overestimated) 되는 경향이 있습니다. 이것은 $q^*(s, a)$, 즉 optimal로 수렴하는데 시간이 더 걸린다.

이것을 보완하기 위해서는 maximization operation을 사용하되 너무 maximum Q값을 선택하지 않아야 한다. 이를 위해서 2개의 Q-table을 만들어서 서로가 서로의 target이 되어서 PE를 진행하는 방법이 있는데 이것을 Double Q-learning이라고 한다.

Q-learning

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

Double Q-learning

$$\begin{cases} Q_1(S,A) \leftarrow Q_1(S,A) + \alpha\left[R + \gamma Q_2(S', \arg\max_a Q_1(S',a)) - Q_1(S,A)\right] \\ Q_2(S,A) \leftarrow Q_2(S,A) + \alpha\left[R + \gamma Q_1(S', \arg\max_a Q_2(S',a)) - Q_2(S,A)\right] \end{cases}$$

TD(λ)

TD $\quad G_t^{(1)} = R_{t+1} + \gamma V(S_{t+1})$

↓

TD(n) $\quad G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})$ 더 많은 step까지 고려하고 싶다

↑

MC $\quad G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-t-1} R_T$

TD(λ)

TD(λ)

TD(n) $$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})$$

n번째 리워드까지 더하는 것은 좋지만 TD(n)은 t + 1 번째부터 t + n - 1 번째 state은 참고하지 않는 문제점이 있다.

$$\begin{cases} G_t^{(1)} = R_{t+1} + \gamma V(S_{t+1}) \\ G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n}) \\ G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-t-1} R_T \end{cases}$$

모든 경우를 다 더해서 모든 state을 참고하자!

$$\longrightarrow \quad G_t^{\lambda} = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

감사합니다