

Value function = state Value function $V_{\pi}(s)$

23/1/23 0 (15)

Value function (가치함수) : policy가 정해져있고, policy를 따라갈 때 각 states를 평가하는 지침
즉, 현재 state s_t 가 어떤 보상을 받는지 나타내는 양.

정의 : - 현재 states에서 policy를 따라갈 때 일어나는 리턴값의 expectation (여러 episode의 리턴값의)
- return G_t 는 에피소드들에서 일어나는 total sum (total discount reward)
- state의 가치 (value)를 평가해주는 함수

$V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$ State-value function $V_{\pi}(s)$

↳ Value function V 는 현재 주어진 π 에서 state s 의 가치를 평가 $V_{\pi}(s)$

↳ 정의 : 현재 state $S_t = s$ 라고 할 때, 현재 state s 에서 출발하여 policy π 를 따라 나올 수 있는 모든 return을 G_t 의 expectation $\mathbb{E}_{\pi}[G_t | S_t = s]$

Value function 배우는 이유 : 현재 state s_t $\xrightarrow{\pi} s'$ next state. action이 좋은지 판별.
↳ a action이 좋다면, s_t 에서 a action을 취했더니 s' 이 더 좋아졌다.

good state 평가하는 방법 : State-Value : $s \xrightarrow{\pi} G_t$ State s 에서 policy π 를 따라간뒤
↳ 다양한 episode의 리턴 G_t 의 기대값이 클 수록 좋은 상태
즉, return들의 expectation을 state의 state-value로 정한뒤.
state-value를 높게 할 수 있는 action의 목표.

가치함수 정의 : 그래서 state-value function을 return들의 expectation으로 정의함

Value function 값이 크면, return들의 expectation이 크고, state에서 앞으로 봄게 될
보상이 커짐

Episode 2 : $s_0, a_0, r_1, s_1, a_1, r_2, s_2, a_2, \dots$

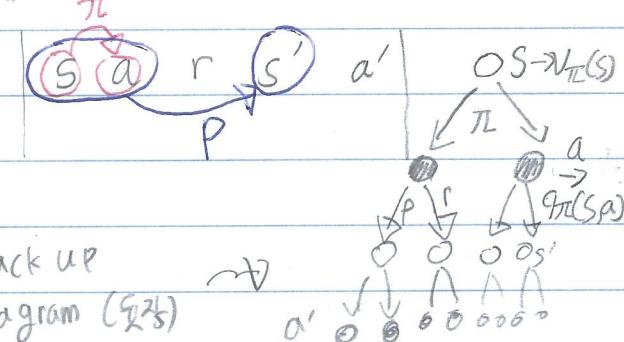
↳ s_0, a_0, r_1 s_0 에서 a_0 를 기반으로 policy π 를 따름 ①

s_0, a_0 pair에서 Next state s_1 으로 가기위해 transition probability P 를 따름 ②

Policy π 와 transition probability P 는 stochastic (랜덤 결과)

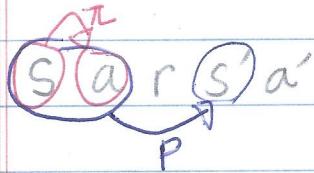
episode 진행 : $s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1}, \dots$

표현 : $s_t, a_t, r_t, s_{t+1}, a_{t+1}$
present state, action
next state, action

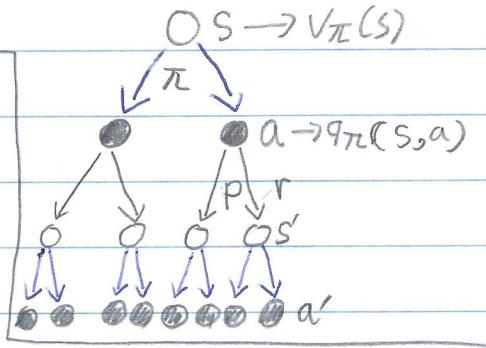


Value functions - state value function $V_{\pi}(s)$

episode



back up diagram



back up diagram explain: 0. s 현재 state에서 출발

present state 3. Policy π 를 따라 action이 결정됨

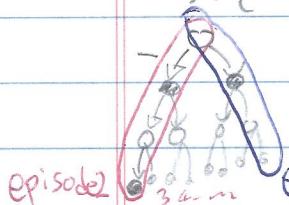
4. action을 취함 (여러 action이 있음)

5. Next state은 transition probability P 에 의해 결정

Next state \rightarrow 6. Next state이 생성됨

7. Next states'에서 policy π 를 따라 next action a' 이 결정됨

- π, P 는 Stochastic이기에 여러가지 경우가 생겨남. 각각의 확률에 따라 랜덤하게 일어짐



episode 2 episode L many episodes. reward들을 sum을 한 total discounted reward를 return of.

$$G_t = r + \gamma \cdot r' + \gamma^2 \cdot r'' + \dots$$

리턴값

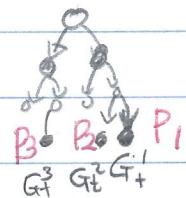
현재 state s 에 있다고 가정. Policy가 π 로 고정. state에서 policy를 따라 액션을 취할 때 앞으로 일어나는 return값의 예측값은? 답: 모든 episode 대한 각 return값의 expectation으로 예측 가능.

Expectation을

구하기: episode가 일어날 확률을 알아야 함. P_1, P_2, \dots 으로 함

$$P^1 \text{ 확률} \times G_t^1 \text{ 리턴} + P^2 \text{ 확률} \times G_t^2 \text{ 리턴} + \dots$$

$$P^1 \cdot G_t^1 + P^2 \cdot G_t^2 + \dots$$



State-value function $V_{\pi}(s)$ 는 주어진 policy π 에 대해 $V_{\pi}(s)$ 로 표현

= policy π 를 따라값을 때 state s 에서부터 시작하여 일어나는 모든 return들의 expectation (expected return)

$V_{\pi}(s) = E[G_t | S_t = s]$: 현재 state s 에서 시작해서 일어나는 모든 return G_t 들의 expectation E_t 이다.

$V_{\pi}(s)$ state s 에 대한 State value function $V_{\pi}(s)$ 이다.

Value functions - Action-value function $q_{\pi}(s, a)$

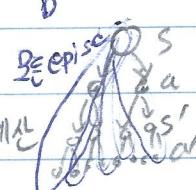
18/163

(state)

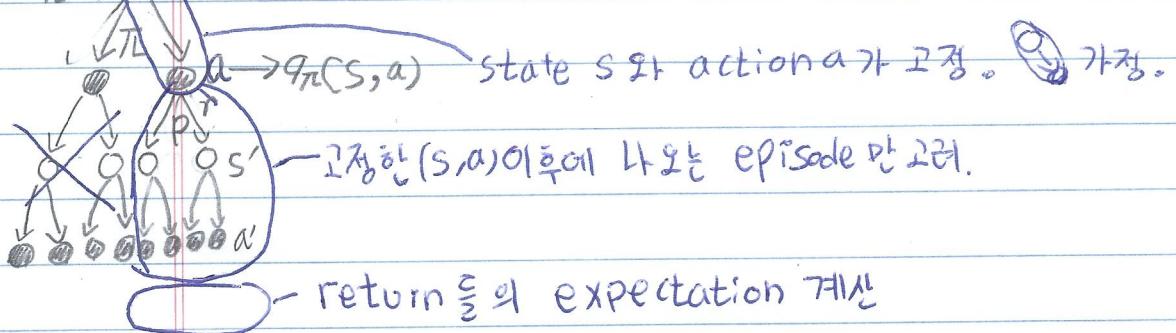
Action-value function $q_{\pi}(s, a)$: 주어진 policy π 에 대해 state s 에서 action a 를 취한 후 시작 expected return을 얻음.

$q_{\pi}(s, a) = \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a]$: $V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$ 와 달리 $A_t = a$ 가 추가되어 있음.
현재 state $S_t = s$ 와 action $A_t = a$ 까지 고정시켜 했기에
 $q_{\pi}(s, a)$ 액션에 추가 되어 있음.

State-value function $V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$ 은 현재 state s 에서 시작하여 나올 수 있는 모든 episode들에 대한 return값들의 expectation을 계산



$s \rightarrow V_{\pi}(s)$ Action-value function $q_{\pi}(s, a)$ 는 state s 와 action a 가 고정되어 있음

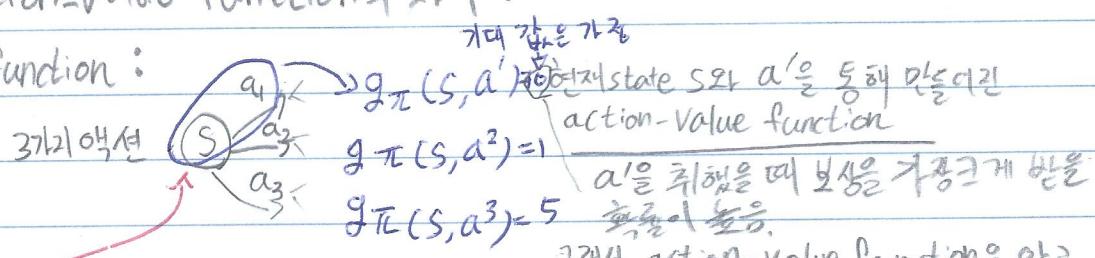


$s \rightarrow V_{\pi}(s)$ State Value function의 시작점

$a \rightarrow q_{\pi}(s, a)$ Action Value function의 시작점

State-Value function과 Action-value function의 차이 :

Action-value function :



Policy 결정 : $V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$ State value function의

값으로는 어떤 action이 좋은지 알 수 없음. 이유 $\rightarrow OS \rightarrow V_{\pi}(s)$

$a \rightarrow q_{\pi}(s, a)$

Action-value function이 훨씬 유용 (가장 큰 장점)

단점 : state-value function은 각 state마다의 값을 계산하면 되지만, action-value function은 각각의 state-action pair에 대해 계산해야 함. 계산해야 할 양이 훨씬 많아짐

Value functions - Dynamic programming

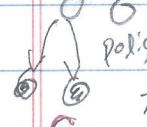
DP와 RL 차이: 1. $V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$, $q_{\pi}(s, a) = \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a]$

expectation \mathbb{E}_{π} 를 계산하는 차이가 있음

DP: Model base, known MDP = transition probability P
를 알고 있음



각 episode가 있을 때, episode가 일어날 확률은



policy π 와 P (transition probability)에서 정해짐.

G_t

각 episode마다 return G_t 가 일어날 확률을 알고 있음. $= V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$
expectation 을 직접 계산할 수 있음.

즉, DP는 $\pi(a|s)$, $P(s'|s, a)$ 를 알고 있기 때문에 \mathbb{E}_{π} expectation 을 직접 계산함.

DP로 expectation 계산: 모든 state에 대한 state-value function $V_{\pi}(s)$ 를 계산하거나

모든 state-action pair에 대한 action value function 을 계산해야 함.

계산량 많음. state-value function $V_{\pi}(s)$ 는 모든 state 만 계산하면 되지만,

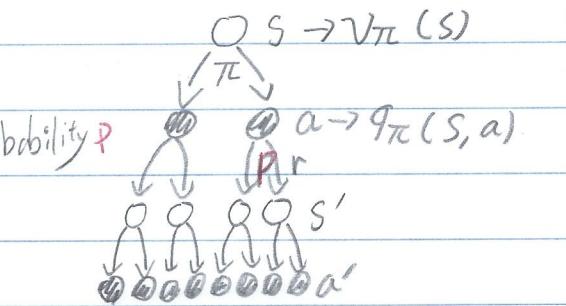
action value function $q_{\pi}(s, a)$ 는 모든 state-action pair 대해 계산해야 함.

action value function $q_{\pi}(s, a)$ 가 훨씬 더 계산량이 많음.

그래서 DP에서는 $q_{\pi}(s, a)$ 를 거의 사용하지 않음

즉, DP는 비교적 숫자가 적은 state에 대해서만 계산하는 state-value function $V_{\pi}(s)$ 에

대해서 적용을 합니다.



* Policy $\pi = s \rightarrow a$ 취할 확률, $P = s \rightarrow s'$ 전이 확률

23/12/31 (1a)

Value functions - reinforcement learning + state value function 18/163
Action-value function

RL :

모델추리, unknown MDP, transition probability P 를
모릅니다.그래서 Action-value function $q_{\pi}(s, a) = \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a]$
expectation \mathbb{E}_{π} 를 제대로 계산할 수 없음.

random samples 를 사용함. random samples >

많아지면 expectation으로 근접함. 이를 이용, 추정치를 계산.

위 방법을 Monte Carlo라고 함.

즉, RL은 Sample로 하기로 모든 state-action pair에 대해서 Action-value function
 $q_{\pi}(s, a)$ 값을 계산하는 것이 아니라, Sample에 있는 것만 계산 후 학습하는 방법.그래서 state action pair $q_{\pi}(s, a)$ 개수가 state $V_{\pi}(s)$ 보다 훨씬 크더라도
RL은 진행할 수 있음.State-Value function $V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$ 와Action-Value function $q_{\pi}(s, a) = \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a]$ 관계식 :

$$V_{\pi}(s) = \sum_a \pi(a|s) q_{\pi}(s, a)$$

해석: 각 Action-value function $q_{\pi}(s, a)$ 와 Action에 대한 확률 $\pi(a|s)$ 을 곱해서Expectation \sum_a 을 계산. 기대값을 계산하면 state-value function 값이 됨* \sum_a 는 모든 가능한 행동 a 에 대해 합산하는 것을 의미합니다.다음과 같이 해석할 수 있습니다. Policy π 에서 상태 s 에서 최선의 행동을 선택할 때,얻을 수 있는 return의 기대값 = \sum_a 모든 가능한 행동에 대한 행동 가치 함수의 기대값예: 1. 상태 s 에서 가능한 행동이 a_1, a_2, a_3 라고 가정2. Policy π 에서 state s 에서 Action a_1 을 선택할 확률 0.5, a_2 확률 0.3, a_3 확률 0.23. 행동 가치 함수 가정 $q(s, a_1) = 10, q(s, a_2) = 15, q(s, a_3) = 20$ State s 에서 Policy π 를 통해 최선의 action을 선택해 얻을 수 있는 보상의 기대값

$$\mathbb{E}[R | s, \pi] = 0.5 \times 10 + 0.3 \times 15 + 0.2 \times 20 = 14$$

이는 모든 가능한 행동에 대한 action-value function의 기대값과 동일합니다.

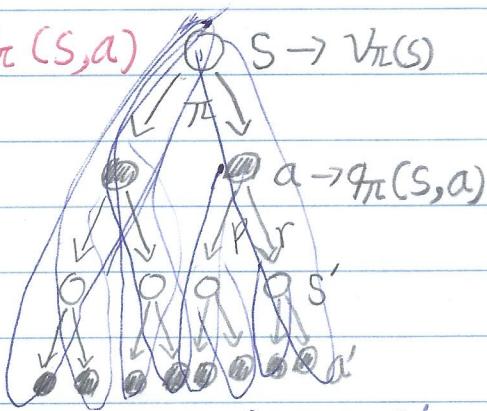
$$\mathbb{E}[Q(s, a) | s, \pi] = 0.5 \times 10 + 0.3 \times 15 + 0.2 \times 20 = 14$$

모든 a 에 대한 a 는 Policy π 에서 state s 에서 최선의 action을 선택해 얻을 수 있는 보상의 기대값모든 a 에 대한 a 는 Policy π 에서 state s 에서 최선의 action을 선택해 얻을 수 있는 보상의 기대값

Value functions - reinforcement Learning

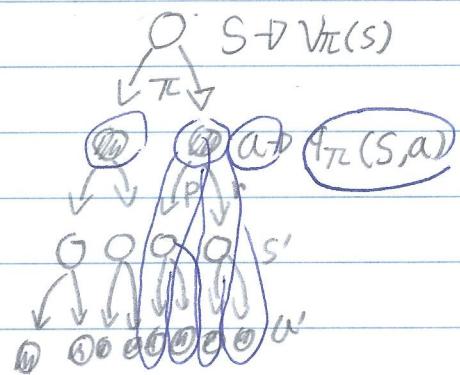
- state value function, advantage function

$$V_{\pi}(s) = \sum_a \pi(a|s) q_{\pi}(s, a)$$



State-Value function $V_{\pi}(s)$

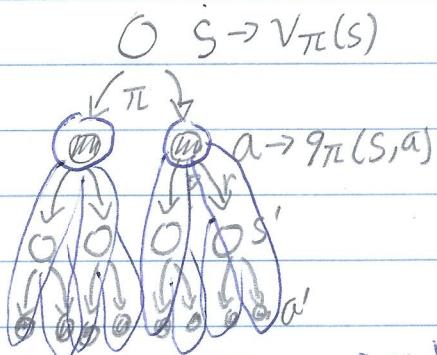
모든 episode에 대한 return들의 expectation



Action - Value function $q_{\pi}(s, a)$

특정한 action 하나하나에 대한

후에 나오는 episode의 return의 expectation



$$\sum_a \pi(a|s) q_{\pi}(s, a) = V_{\pi}(s)$$

Action value function $q_{\pi}(s, a)$ 를 모든 액션에 대해 expectation을 취하면 State-value function과 같다.

현재 states에서 여러 action 중 특정 action이 평균보다 좋은 액션일 확률

Advantage function $A_{\pi}(s, a) = q_{\pi}(s, a) - V_{\pi}(s)$: 나쁜지 평가하기 위한 함수

$A_{\pi}(s, a)$ 는 State-action pair에 대한 함수. Action-value function $q_{\pi}(s, a)$ 와 State-value function $V_{\pi}(s)$ 과의 차이.

$$A_{\pi}(s, a)$$

$$q_{\pi}(s, a) - \sum_a \pi(a|s) q_{\pi}(s, a)$$

의미: state-value function $V_{\pi}(s)$ 는 모든 action에 대한 action-value function의 평균
특정한 action에 대한 action-value function에서 모든 action에 대한 action-value function을 뺀다.

$$A_{\pi}(s, a) = q_{\pi}(s, a) - \sum_a \pi(a|s) q_{\pi}(s, a) = q_{\pi}(s, a) - \cancel{q_{\pi}(s, a)} - V_{\pi}(s)$$

$q_{\pi}(s, a) - V_{\pi}(s) > 0$ 이라면, 특정한 action a $q_{\pi}(s, a)$ 가 평균보다 좋음 a : 좋은

$q_{\pi}(s, a) - V_{\pi}(s) < 0$ $q_{\pi}(s, a)$ action a 가 평균보다 $V_{\pi}(s)$ 가 더 좋은 적용

a : 나쁜 action

확률이론 - Law of total probability

전체 확률

$\cup = \text{Union}$
 $\cap = \text{intersection}$

05 Bellman equation

24/01/01 (2)

9/16/3

의 법칙

Union

모든 B_n 들이 서로 \cap 하며 즉 동시에 일어날 수 있다.

Law of total probability (전체 확률의 법칙)

$\cdot P(A) = \sum_n P(A \cap B_n) \cdot P(B_n)$: B_n 의 두 가지 조건 1. $\cup B_n$ 전체, 2. $B_m \cap B_n = \emptyset$

$$P(A) = P(A \cap (\cup B_n)) = \sum_n P(A \cap B_n)$$

조건부 확률 변환

$B_n = B_{1:n}$

설명 : $P(A) = \sum_n P(A \cap B_n)$ 는 사건 A가 각각의 B_n 과 함께 발생 확률을 모두 더한 후

그러나, 사건 A가 B_n 과 동시에 발생하는 확률 $P(A \cap B_n)$ 은 다를 수 있음.

간단한 확률로 표기하기 위해 조건부 확률 사용 : $P(A \cap B_n) = P(A | B_n) \cdot P(B_n)$

$\hookrightarrow P(A | B_n)$ 은 B_n 이 일어날 A의 조건부 확률, $P(B_n)$ 은 마지막 확률입니다.

$P(A) = \sum_n P(A | B_n) \cdot P(B_n)$ 식을 통해 복잡한 사건 A의 확률을 보다 간단한

조건부 확률과 마지막 확률의 곱의 합으로 나타낼 수 있게 됨.

Random variable X에 대한 expectation

$$\circ E[X] = \sum_y E[X | Y=y] \cdot P(Y=y) \quad \text{Note } E[X] = \sum_x P(X=x) \cdot x, \quad \text{for } x = \text{모든 경우}$$

$$E[X | Z=z] = \sum_x x \cdot P(X=x | Z=z)$$

- $E[X] = \sum_x x \cdot P(X=x)$ 설명 : - \sum_x 모든 x에 대해 합함.
- $P(X=x)$ 는 확률 변수 X가 값을 취할 확률.

\hookrightarrow 확률 변수 X에 대한 기대값 (expectation) $E[X]$ 는 확률 변수 X가 취할 수 있는 모든 값을

각각의 확률로 가중 평균한 값

\hookrightarrow 이 기대값 공식은 이산 확률 변수 (discrete random variable)에 대한 것.

이산 확률 변수는 세 수 있는 값들을 가질 수 있는 변수이며, 기대값은 각 값을 값이 발생할 확률로 가중한 후 모두 더하여 계산합니다.

예 : 주사위 굴리기 \rightarrow 눈금이 1부터 6까지 있고, 이산 확률 변수 X의 기대값 \rightarrow []

$$\rightarrow E[X] = (1/6)x1 + (1/6)x2 + (1/6)x3 + (1/6)x4 + (1/6)x5 + (1/6)x6 = 3.5$$

\rightarrow 여러 번 주사위를 던졌을 때 눈금의 평균값은 약 3.5 값을 기대할 수 있을

+ 연속 확률 변수 (continuous random variable)의 기대값은 적분을 사용함

전체 확률이론 - Law of total probability

24/01/01 (22)

16:12~16:41 a/163

Note: $P(A) = \sum_n P(A|B_n) \cdot P(B_n)$

~~E[X]~~: $E[X] = \sum_y E[X|Y=y] \cdot P(Y=y)$ (Note $E[X] = \sum_x x P(X=x)$)

$E[X|Z=z] = \sum_x x P(X=x|Z=z)$

$E[X]$ 유도: $E[X] = \sum_x x P(X=x)$ 예 $P(A) = \sum_n P(A|B_n) \cdot P(B_n)$ 6/2 적용

$E[X] = \sum_x x \sum_y P(X=x|Y=y) \cdot P(Y=y)$, $Y=y$ 는 B_n 을 나타냄

Summation 위치변경: $= \sum_y \sum_x x \cdot P(X=x|Y=y) \cdot P(Y=y)$, $\sum_x x \cdot P(X=x|Y=y)$ 를 $E[X|Y=y] = \sum_x x P(X=x|Y=y)$ 로 대체

↓

로 대체

$|Z=z$

expectation 으로 변경: $= \sum_y E[X|Y=y] \cdot P(Y=y)$

즉, $E[X] = \sum_y E[X|Y=y] \cdot P(Y=y)$ 을 유도함.

조건부 $Z=z$ 추가

$E[X|Z=z] = \sum_y E[X|Y=y, Z=z] \cdot P(Y=y|Z=z)$

총정리 Law of total probability 전체 확률의 법칙

- $P(A) = \sum_n P(A|B_n) \cdot P(B_n)$
- $E[X] = \sum_y E[X|Y=y] \cdot P(Y=y)$
- $E[X|Z=z] = \sum_y E[X|Y=y, Z=z] \cdot P(Y=y|Z=z)$

Note that $E[X] = \sum_x x P(X=x)$

$E[X|Z=z] = \sum_x x P(X=x|Z=z)$

Law of large numbers

랜덤 Variable X 의 Expectation $E[X]$ 를 계산해야하나 실제 계산이 어려워
랜덤 Sample $\bar{X} = \frac{1}{n}(X_1 + \dots + X_n)$ 으로 계산을 향. Random Sample의 수가 커지면, 점점
Expectation $E[X]$ 의 값으로 근접.

$$\bar{X} = \frac{1}{n}(X_1 + \dots + X_n) \rightarrow E[X] \text{ as } n \rightarrow \infty$$

예: Coin tossing (동전 던지기) H(앞): 1정, T(뒤): 0정
bias 확률 P , $1-P$

기대값 X 는?

$$E[X] = 1 \cdot P + 0 \cdot (1-P) = P \leftarrow \text{앞면이 나올 확률을 알고 있으면 기대값 } X \text{를 알}$$

만약 P 를 모르면?

여러번 동전을 던져서 유추하기. random samples (trials)

랜덤 samples 10번 \rightarrow 앞면 H 6번 $\rightarrow P \approx 0.6$

trials 100번 \rightarrow H 585 $\rightarrow P \approx 0.585$

정의: random samples (trials)를 많이 시도하여 얻어지는 결과의 평균이 expected value에 근접
trials 수가 많을수록 expected value에 더 근접함.

1. $X = X_1, \dots, X_n$ i.i.d random samples (independent and
- identically distributed 독립 항등 분포), X 들은 독립적 예) 첫 번째 동전, 두 번째 동전 독립
identically distributed는 X 들은 같은 확률분포, 예) 첫 번째, 두 번째 동전의 확률 같음.

2. $\bar{X} = \frac{1}{n}(X_1 + \dots + X_n) \rightarrow E[X] \text{ as } n \rightarrow \infty$

샘플들 평균, 표본 평균

표본 평균값은 n 이 점점 커질 때 $E[X]$ 로 다가간다.

사용 이유: 강화학습시 expectation을 계산하지 않고 극치를 사용하는 근거

Bellman Equation - $V_{\pi}(s)$

24/01/02 (24)

10:30 ~ 11:15 20/163
2:00 ~ 11:25 ~ 45Bellman expectation equation: $V_{\pi}(s)$, $q_{\pi}(s, a)$ 에 대해 2가지 식이 있다.State Value function 정의 $V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$ G_t 리턴값을 알기 위해서는 episodes가 끝날 때까지 기다려야 함. Game이 끝날 때까지는 $V_{\pi}(s)$ 를 계산
 $G_t \leftarrow \text{return} \leftarrow \text{episodes}$ 종료law of total probability를 통해 $V_{\pi}(s)$ 변형: $\mathbb{E}[X | Z=z] = \sum_y \mathbb{E}[X | Y=y, Z=z] \cdot P(Y=y | Z=z)$

$$V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s] = \mathbb{E} \sum_a (\mathbb{E}_{\pi}[G_t | S_t = s, A_t = a] \cdot P(A_t = a | S_t = s))$$

$$= (\sum_a \pi(a | s) q_{\pi}(s, a))$$

$$= \sum_a \pi(a | s) \mathbb{E}_{\pi}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a]$$

$$G_t \rightarrow R_{t+1} + \gamma G_{t+1} \text{ 유도: return } G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots = R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \dots)$$

$$= R_{t+1} + \gamma G_{t+1}$$

$$= \sum_a \pi(a | s) \sum_{s', r} \mathbb{E}_{\pi}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a, S_{t+1} = s', R_{t+1} = r]$$

$$\cdot P(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a)$$

 $\mathbb{E}_{\pi} \rightarrow \sum_{s', r} \mathbb{E}_{\pi}$: Next states' 및 reward 부분이 sum으로 분리되었음.Next states' 을 분리하면서, $S_{t+1} = s'$ 조건부에 들어감, 확률을 곱함Reward \rightarrow sum으로 분해하면서, $R_{t+1} = r$ 조건부에 들어감, 확률에 곱함

Law of total probability의 2가지 (Next state와 Reward)에 대해 sum으로 분해

present state-action pair에서 next states'와 reward가 나올 확률

$$= \sum_a \pi(a | s) \sum_{s', r} P(s', r | s, a) [r + \gamma \mathbb{E}_{\pi}[G_{t+1} | S_{t+1} = s']]$$

∴ G_{t+1} is independent on S_t, A_t in MPP

$$\mathbb{E}_{\pi}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a, S_{t+1} = s', R_{t+1} = r] = [r + \gamma \mathbb{E}_{\pi}[G_{t+1} | S_{t+1} = s']]$$

1. $R_{t+1} = r$ 고정, R_{t+1} 은 r 로 변환 가능. 즉 random variable R_{t+1} 에서 특정값 r 로 변환. r 은 \mathbb{E}_{π} 로 나온다.2. discount vector γ 도 상수여서, expectation 밖으로 나올 수 있음.3. expectation \mathbb{E} 안에는 G_{t+1} 과 $S_{t+1} = s'$ 만 조건부로 남아 있음4. $A_t = a, S_{t+1} = s'$ 이 사라진 이유: $S_{t+1} = s'$ 을 present state로 간주. $t+1 = \text{present}, t = \text{past}$ MDP에서는 present state G_{t+1} 이후에 벌어진 일은 past state S_t, A_t 에 독립적임.

return은 present state 이후부터 일의, past state은 독립적임, 조건부에 영향이 없음, 제거 가능

Bellman Equation - $V_{\pi}(s)$

Bellman expectation equation $V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$ 에 대해서 식을 나열. (전체이기)

$$\begin{aligned}
 V_{\pi}(s) &= \mathbb{E}_{\pi}[G_t | S_t = s] \\
 &= \sum_a \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a] \cdot P(A_t = a | S_t = s) (= \sum_a \pi(a | s) q_{\pi}(s, a)) \\
 &= \sum_a \pi(a | s) \mathbb{E}_{\pi}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a] \\
 &= \sum_a \pi(a | s) \sum_{s', r} \mathbb{E}_{\pi}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a, S_{t+1} = s', R_{t+1} = r] \\
 &\quad \cdot P(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a) \\
 &= \sum_a \pi(a | s) \sum_{s', r} P(s', r | s, a) [r + \gamma \mathbb{E}_{\pi}[G_{t+1} | S_{t+1} = s']] \because G_{t+1} \text{ is independent on } S_t, A_t \text{ in MDP} \\
 &= \sum_a \pi(a | s) \sum_{s', r} P(s', r | s, a) [r + \gamma V_{\pi}(s')] (= \sum_a \pi(a | s) [R_s^a + \gamma \sum_{s'} P_{ss'}^a V_{\pi}(s')]) \\
 &= \mathbb{E}_{\pi}[R_{t+1} + \gamma V_{\pi}(S_{t+1}) | S_t = s]
 \end{aligned}$$

이 페이지에서
진행한 내용

$$\begin{aligned}
 \text{빨간색 설명} &= \sum_a \pi(a | s) \sum_{s', r} P(s', r | s, a) [r + \gamma \mathbb{E}_{\pi}[G_{t+1} | S_{t+1} = s']] \\
 &= \sum_a \pi(a | s) \sum_{s', r} P(s', r | s, a) [r + \gamma V_{\pi}(s')] \quad \leftarrow \begin{array}{l} \text{present} \\ \downarrow \\ \text{2번째 줄} \\ V_{\pi}(s) \\ = \mathbb{E}_{\pi}[G_t | S_t = s] \\ S \in S'(\text{present}) \text{로 확장} \end{array}
 \end{aligned}$$

$$\begin{aligned}
 \text{빨간색 설명 2} &= \sum_a \pi(a | s) \sum_{s', r} P(s', r | s, a) [r + \gamma V_{\pi}(s')] \\
 &\rightarrow \text{현재 state } s \text{에서 } \pi \text{ policy 와 transition probability } P \text{를 이용해서 얻어지는 } a, r, s' \\
 &\rightarrow \textcircled{2} \rightarrow a, r, s'
 \end{aligned}$$

$$\begin{aligned}
 &\pi(a | s), P(s', r | s, a) : 확률을 곱해준 의미?; a, r, s'가 나올 확률을 계산함. \rightarrow \mathbb{E}_{\pi} \\
 &= \mathbb{E}_{\pi}[R_{t+1} + \gamma V_{\pi}(S_{t+1}) | S_t = s]
 \end{aligned}$$

$$\begin{aligned}
 \text{설명 2-1} &= [r + \gamma V_{\pi}(s')] \quad \leftarrow \sum_a \pi(a | s) \sum_{s', r} P(s', r | s, a) [r + \gamma V_{\pi}(s')] \\
 &= [R_{t+1} + \gamma V_{\pi}(S_{t+1}) | S_t = s] \quad \leftarrow = \mathbb{E}_{\pi}[R_{t+1} + \gamma V_{\pi}(S_{t+1}) | S_t = s]
 \end{aligned}$$

Random Variable R_{t+1}, S_{t+1} 에 대한 확률 $\pi(a | s), P(s', r | s, a)$ 로 풀려있음

R_{t+1} 의 reward r 을 적어둔것이고, S_{t+1} 에는 next state s' 을 넣어둔것, 그래서 같은식임.

Bellman expectation equation 결론: State Value function $V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$

State Value function 변형: $\mathbb{E}_{\pi}[R_{t+1} + \gamma V_{\pi}(S_{t+1}) | S_t = s]$ ← Bellman expectation식

Bellman expectation equation: State Value function $V_{\pi}(s)$ 을 decomposing (분해)하여

Recursive equation (재귀식)으로 바꿈. 어떤식으로?: $V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$ 을 return G_t 대입

immediate(즉각) reward R_{t+1} 과 discount next state value $\gamma V_{\pi}(S_{t+1})$ 의 sum으로 분해 가능

분해 시장점: $V_{\pi}(s)$ 를 계산하기 위해 더이상 return의 필요하지 않음. (return G_t) R_{t+1}, S_{t+1} 만 알면됨. episode 전체가 끝나기 전 계산됨.

Bellman Equation - $q_{\pi}(s, a) + V_{\pi}(s)$

$$q_{\pi}(s, a) = \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a]$$

$$= \sum_{s', r} p(s', r | s, a) [r + \gamma \sum_a \pi(a' | s') q_{\pi}(s', a')]$$

$$= \mathbb{E}_{\pi}[R_{t+1} + \gamma q_{\pi}(S_{t+1}, A_{t+1}) | S_t = s, A_t = a] \leftarrow \text{Bellman expectation } \leftarrow q_{\pi}(s, a)$$

$$V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s] = \sum_{s', r} p(s', r | s, a) [r + \gamma V_{\pi}(s')]$$

$$q_{\pi}(s, a) = \mathbb{E}_{\pi}[G_t | S_t = s, A_t = a] \quad A_t = a \text{ 만 } \rightarrow \text{가}$$

$$V_{\pi}(s) = \sum_a \pi(a | s) \sum_{s', r} p(s', r | s, a) [r + \gamma V_{\pi}(s')] \quad \pi(a | s) \text{은 action } a \text{로 고정되어 있어 } \gamma \text{은}$$

$$q_{\pi}(s, a) = \sum_{s', r} p(s', r | s, a) [r + \gamma \sum_a \pi(a' | s') q_{\pi}(s', a')] \quad \leftarrow V_{\pi}(s) = \sum_a \pi(a | s) q_{\pi}(s, a) \text{ 공식 (앞)}$$

$$V_{\pi}(s) = \mathbb{E}_{\pi}[R_{t+1} + \gamma V_{\pi}(S_{t+1}) | S_t = s] \quad \leftarrow V_{\pi}(S_{t+1}) \text{은}$$

$$q_{\pi}(s, a) = \mathbb{E}_{\pi}[R_{t+1} + \gamma q_{\pi}(S_{t+1}, A_{t+1}) | S_t = s, A_t = a] \quad \text{Action value function } q_{\pi}(s, a) \text{로 } \leftarrow$$

$$\text{정리 } V_{\pi}(s) = \sum_a \pi(a | s) \sum_{s', r} p(s', r | s, a) [r + \gamma V_{\pi}(s')] (= \sum_a \pi(a | s) [R_s^a + \gamma \sum_{s'} p_{ss'}^a V_{\pi}(s')])$$

$$\text{Slide 14 page 87} \quad 1. P_{ss'}^a = p(s' | s, a) = p(S_{t+1} = s' | S_t = s, A_t = a) = \sum_{r \in R} p(s', r | s, a)$$

$$2. R_s^a = r(s, a) = \mathbb{E}[R_{t+1} | S_t = s, A_t = a] = \sum_{r \in R} r \sum_{s' \in S} p(s', r | s, a)$$

$$\sum_{s', r} p(s', r | s, a) [r] \quad \text{Reward } r \text{은 모든 리워드 } r \text{과 Next state } s' \text{에 대해서 확률 } p(s', r | s, a) \text{을 곱한 } \text{sum}$$

$$2. R_s^a = \sum_{r \in R} r \sum_{s' \in S} p(s', r | s, a) \quad : \text{reward } r \text{과 확률 } p(s', r | s, a) \text{를 } R \text{의 reward } r \text{과} \\ \text{next state pairs } s' \text{에 대해서 sum 해요. } R = \{a_1, a_2, \dots, a_n\} \quad R_s^a$$

$$\sum_{s', r} p(s', r | s, a) [r V_{\pi}(s')] = \gamma \sum_{s'} p_{ss'}^a V_{\pi}(s') \quad : \sum_r p(s', r | s, a) 시메이션의 리워드만 적용되는 부분으로$$

$$1. P_{ss'}^a = \sum_{r \in R} p(s', r | s, a) \quad : \text{리워드에 대해서만 sum}$$

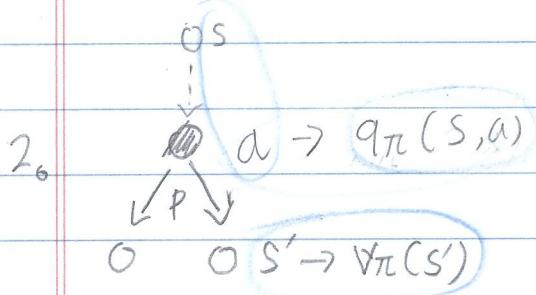
Bellman equation - Backup Diagram

$$6. \quad S \rightarrow V_{\pi}(s)$$



$$a \rightarrow q_{\pi}(s, a)$$

식1 $V_{\pi}(s) = \sum_a \pi(a|s) q_{\pi}(s, a)$: $q_{\pi}(s, a)$ 을 모든 action에 대해 적용하면 $V_{\pi}(s)$ 얻



1. a action을 시작 next state s' 으로 넘어가는 때

2. a action은 present state에서 넘어온다

3. state s - action a pair에 대한

$$\text{식2 } q_{\pi}(s, a) = R_s^a + \gamma \sum_s P_{ss'}^a V_{\pi}(s')$$

Action-value function $q_{\pi}(s, a)$ 하고

next state s' 에 대한 $V_{\pi}(s')$ state value function

고려를 확장

4. Action value function $q_{\pi}(s, a)$ 는 $\sum_{s'} q_{\pi}(s', a)$, action에서 나오는 모든 episode에 대한
리턴들의 expectation

5. Next state에 대한 state-value function $V_{\pi}(s')$ 는 $\sum_{s'} R_{s'}^s$, s' 이후에 나오는 모든 episodes에
대한 return들의 expectation

6. 즉, $q_{\pi}(s, a)$ 는 각각 next states에 대해서

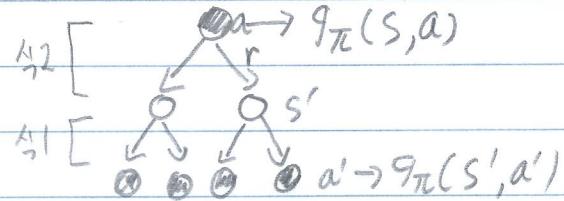
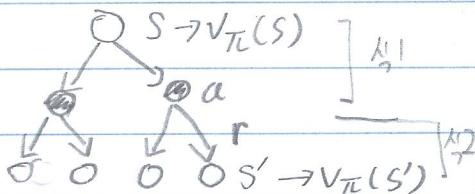
$V_{\pi}(s')$ 을 expectation 계산한 결과 같음.

7. $q_{\pi}(s, a)$ 를 계산하기 위해 $\sum_s P_{ss'}^a V_{\pi}(s')$ 이 계산되어야 함.

8. $\sum_{s'} R_{s'}^s$, 한 step을 넘어갈 때마다 discount 하기로 함

9. $\sum_{s'} R_{s'}^s$ \downarrow a 에서 s' 으로 가는 중간 일어진 reward도 sum.

3.



식3

$$V_{\pi}(s) = \sum_a \pi(a|s) [R_s^a + \gamma \sum_s P_{ss'}^a V_{\pi}(s')]$$

= 식1의 $q_{\pi}(s, a)$ 를 식2로 치환

$V_{\pi}(s)$ 와 $V_{\pi}(s')$ 의 관계식

$q_{\pi}(s, a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a \sum_a \pi(a|s') q_{\pi}(s', a')$

= 식2의 $V_{\pi}(s')$ 를 식1으로 치환 ($s \rightarrow s'$, $a \rightarrow s'$ 으로 변경)

$q_{\pi}(s, a)$ 와 $q_{\pi}(s', a')$ 의 관계식

24/01/03 (28)

12:30 ~ 22/163

13:20

Optimal Value functions and Policy

optimal policy

강화학습의 목적: reward를 최대로 할 수 있는 policy 찾기, 전망 가치인 Value function을 찾았을 때

이번강의: 1. Optimal policy를 찾기 위한 Optimal Value functions

- Optimal state-value function 과 optimal action-value function

2. 위를 찾기 위해 필요한 Bellman optimality equation.

Optimal Value functions and Policy

Optimal value function (최적 가치함수): Value functions 사이에서 maximum value.

optimal value functions을 이용하여 optimal policy를 찾게 되고,

optimal policy를 찾는 것이 MDP에서의 궁극적인 목표.

MDP가 solved? 해하는 것 optimal value functions을 찾았고, 이를 이용하여 optimal policy를 찾을 때

Optimal state-value function $V_*(s) = \max_{\pi} V_{\pi}(s)$ 각 state s 에 대해서 state-value function $V_{\pi}(s)$ 값이 maximum이 되는 policy π 를 적용한 값. (Optimal = *)Optimal action-value function $q_*(s, a) = \max_{\pi} q_{\pi}(s, a)$ 각 state-action pair (s, a) 에 대해서, Action value function q_{π} 값을 maximized하는 action π 가 적용되어 들어진 값. < optimal policy를 찾는데 꼭 필요한 Value function고려해볼 문제 state마다 maximum이 되는 policy π 가 다르다면?예: $V_{\pi_1}(s') > V_{\pi_2}(s')$, $V_{\pi_1}(s^2) < V_{\pi_2}(s^2)$ 설명: 1. Policy π 는 state-value function 값이 높을 수록 좋은 것, '기대 할 수 있는 리턴값이 더 큼 policy π 사이에 우선순위를 주는 ordering을 할 수 있음.Policy π', π 사이의 order를 정하는 기준: 모든 state s 에 대해서 π' 를 이용한 state-value function $V_{\pi'}(s)$ 값이 π 를 이용한 state-value function $V_{\pi}(s)$ 값보다 항상 크거나 같으면 π' 가 π 보다 더 좋다.즉, 모든 s 에 대해 $V_{\pi'}(s) \geq V_{\pi}(s)$ 면 π' $\geq \pi$ 다. Ordering policies $\pi' \geq \pi$ if $V_{\pi'}(s) \geq V_{\pi}(s)$ for all s . 그러나 이를 이용하여 'maximum policy π 가 다른 상황'의 order는 정할 수 있음.

해결방법

Optimal policy $\pi_* \geq \pi$ for all π .'π', π 사이의 order를 정하는 기준'과 같이 고려하면 $V_{\pi_*}(s) \geq V_{\pi}(s)$ for all s , for all π 그러면 state마다 최대 policy가 다른 상황은 생기지 않음. 즉 $V_{\pi_1}(s^1) \geq V_{\pi_2}(s^1)$, $V_{\pi_1}(s^2) < V_{\pi_2}(s^2)$ 모든 state에 대해서 하나의 optimal policy π_* 로 적용 가능. $V_{\pi_*}(s) = \max_{\pi} V_{\pi}(s)$, $q_{\pi_*}(s, a) = \max_{\pi} q_{\pi}(s, a)$

증명

MDP는 optimal policy π_* 가 항상 존재. (적어도 한개) $V_{\pi_*}(s) = V_*(s)$, $q_{\pi_*}(s, a) = q_*(s, a)$

Finding an optimal policy

06 Bellman equation 2

24/01/03 (29)

1:55 2:45

23/163

Finding an optimal policy: optimal policy를 알고 있다면 $q_*(s, a)$ 를 maximizing하는 액션 찾기.

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \arg \max_a q_*(s, a) \\ 0 & \text{otherwise} \end{cases}$$

현재 주어진 state s 에서 어떤 action a 를 취하는 것이 가장 큰 리워드를 얻는지.

- Optimal action value pair $q_*(s, a)$ 값을 모든 state-action pair에 대해 알고 있음

- 어떤 action a 를 취했을 때 $q_*(s, a)$ 가 가장 높을 수 있는 방법

↳ $q_*(s, a)$ 을 최대로 하는 action a 가 가장 좋은 액션

그래서 action a 를 선택하면 optimal policy를 얻을 수 있음. 1을 주고, 나머지는 0을 주고 100% 선택

즉, $q_*(s, a)$ 를 찾을 수만 있으면, optimal $\pi_*(s) = \arg \max_a q_*(s, a)$

optimal policy π_* 는 주어진 state s 에서 $q_*(s, a)$ 값을 최대로 하는 a 를 취하는 것.

State가 주어지면 action이 결정되므로 deterministic policy (Non-stochastic)

MDP에서는 deterministic optimal policy가 항상 존재 (앞장 맨 아래를)

참고사항

DP 초점 식 1 $V_*(s) = \max_a q_*(s, a)$ by $V_{\pi}(s) = \sum_a \pi(a|s) q_{\pi}(s, a)$

설명: $V_{\pi}(s) = \sum_a \pi(a|s) q_{\pi}(s, a)$ 는 05 Bellman Equation 1에서 얻었음.

Optimal로 변경하면 $V_{\pi_*}(s) = \sum_a \pi_*(a|s) q_{\pi_*}(s, a)$ ↳ $\sum_a \pi_*(a|s) = \arg \max_a q_*(s, a)$ 로 변경 가능
앞장의 규칙 $V_*(s) = \max_a q_*(s, a)$ ↳ 식의 모든 $q_*(s, a)$ 를 최대로 하는 $\arg \max_a$ 액션만 선택 + $q_*(s, a)$ 를 max로

즉, $\sum_a \pi_*(a|s) q_{\pi_*}(s, a)$ 가 $\max_a q_*(s, a)$ 로 바뀜 (증명)

즉, $q_*(s, a)$ 를 알고 있다면 $V_*(s)$ 를 계산할 수 있음.

05 강의

RL 초점 식 2 $q_*(s, a) = r(s, a) + \gamma \sum_{s'} p(s'|s, a) V_*(s')$ by $q_{\pi}(s, a) = r(s, a) + \gamma \sum_{s'} p(s'|s, a) V_{\pi}(s')$

설명: $V_*(s)$ 를 알고 있다면 $q_*(s, a)$ 를 계산할 수 있음.

조건 1. transition probability $p(s'|s, a)$ 를 알아야 계산 가능 + reward도 알아야 함 $r(s, a)$

즉, $V_*(s')$ 에서는 $q_*(s, a)$ 를 직접적으로 얻을 수 있기 때문. $p(s'|s, a)$ 를 알아야 함, Model-based 일 때 가능, Non-MDP 일 때 가능

이 장의 결론: optimal policy $\pi_*(s) = \arg \max_a q_*(s, a)$ ↳ $\pi_*(s)$ 를 계산하기 위해서는 $q_*(s, a)$ 를 알아야 함.

(참고사항) $q_*(s, a)$ 를 알면 $V_*(s)$ 를 쉽게 찾을 수 있지만, $V_*(s)$ 를 알아도 $p(s'|s, a)$ 알아야 $q_*(s, a)$ 찾을 수 있음.

DP에서는 transition probabilities p 를 알고 있기에, $V_*(s)$ 와 $q_*(s, a)$ 무언가 찾던 π_* 를 알 수 있음.

Model-free의 경우 p 를 모르면, $V_*(s)$ 로 $q_*(s, a)$ 찾을 수 없음. 반드시 $q_*(s, a)$ 를 처음부터 찾아야 π_* 알 수 있음.

$q_*(s, a)$ 찾는 방법: random samples 이용해서 $q_*(s, a)$ 를 approximate하는 Q(s, a)에 대해서 Q-values를 계산하는 방법이 Reinforcement learning임

Bellman optimality equation

06 Bellman equation

24/01/03 (30)

3:45~4:35 24/163

$$V_*(s) = \max_{a \in A(s)} q_*(s, a) = \max_a \mathbb{E}_{\pi_*} [r_{t+1} + \gamma V_*(s_{t+1}) | S_t = s, A_t = a]$$

설명 $q_*(s, a)$ 은 $q_{\pi_*}(s, a)$ 로 변경, $q_{\pi_*}(s, a) = \mathbb{E}_{\pi_*} [G_{t+1} | S_t = s, A_t = a]$

$$= \max_a \mathbb{E}_{\pi_*} [R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a]$$

설명 $G_{t+1} \triangleq R_{t+1} + \gamma G_{t+2}$ 05정승, $\mathbb{E}_{\pi_*} [G_{t+1} | S_t = s, A_t = a]$

$$= \max_a \mathbb{E}[R_{t+1} + \gamma V_*(s_{t+1}) | S_t = s, A_t = a]$$

설명 $\mathbb{E}_{\pi_*} [G_{t+1}] \triangleq \mathbb{E}[V_*(s_{t+1})]$

$$= \max_a \sum_{s', r} p(s', r | s, a) [r + \gamma V_*(s')]$$

설명 $\mathbb{E}_{\pi_*} [G_{t+1}]$ 은 리턴에 적용되어, 예외로 끝날 때까지 적용

그리고 $\mathbb{E}[R_{t+1} + \gamma V_*(s_{t+1})]$ 는 immediate reward R_{t+1} 과 S_{t+1} Next state까지 적용

그리고 \mathbb{E} 를 sum으로 분해하여 $\sum_{s', r}$ immediate reward와 next state까지 적용

그때 확률 $p(s', r | s, a)$ 인데, $\mathbb{E}[R_{t+1} + \gamma V_*(s_{t+1})]$ 을 그대로 옮기 $[r + \gamma V_*(s')]$ 을 만듬

$$\max_a \sum_{s', r} p(s', r | s, a) [r + \gamma V_*(s')] \leq V_{\pi}(s) = \sum_a \pi(a | s) \sum_{s', r} p(s', r | s, a) [r + \gamma V_*(s')]$$

설명: $\sum_a \pi(a | s)$, V_{π} 가 π_* optimal policy로 바뀜 $\max_a V_*$, 나머지는 같음

$$= \max_a [R_s^a + \gamma \sum_{s'} P_{ss'}^a V_*(s')]$$

$$q_*(s, a) = \mathbb{E}[R_{t+1} + \gamma \max_{a'} q_*(s_{t+1}, a') | S_t = s, A_t = a]$$

설명 π 대신 π_* optimal로 바뀜, maximize 시키는 a' 으로 바뀜 $\max_{a'}$

$$= \sum_{s', r} p(s', r | s, a) [r + \gamma \max_{a'} q_*(s', a')]$$

설명 $\max_{a'} q_*(s', a')$ 이 Bellman expectation에서는 $\sum_{a'} \pi(a' | s') q_*(s', a')$ 이 있음

$\pi \neq \pi_*$, $\sum_{a'} \pi(a' | s') \rightarrow \max_{a'}$. 나머지는 동일.

$$= R_s^a + \gamma \sum_{s'} P_{ss'}^a \max_{a'} q_*(s', a')$$

Bellman optimality equation: $V_*(s) = \max_a \sum_{s', r} p(s', r | s, a) [r + \gamma V_*(s')]$

$$q_*(s, a) = \sum_{s', r} p(s', r | s, a) [r + \gamma \max_{a'} q_*(s', a')]$$

$p(s', r | s, a)$ 과 r 을 알고 고정. \rightarrow model-based (known MDP). 이런 경우

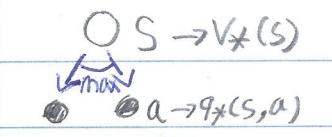
위 공식을 반복 사용하여 $V_*(s)$, $q_*(s, a)$ 를 알 수 있음 (functions can be iteratively computed)

찾는 방법이 Dynamic Programming임.

대부분 DP에서는 모든 state-action pair마다 $q_*(s, a)$ 를 찾아가는 방식, state에 대해서 $V_*(s)$ 가 끝난다, DP는 주로 $V_*(s)$ 를 사용

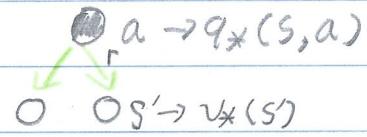
Bellman optimality equation의 Backup diagram

06 Bellman Equation
24/01/03 (31)
25/1/63



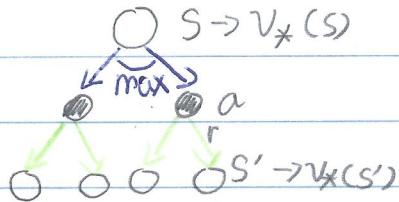
$$V_*(s) = \max_a q_*(s, a)$$

(cf. $V_\pi(s) = \sum_a \pi(a|s) q_\pi(s, a)$)



$$q_*(s, a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a V_*(s')$$

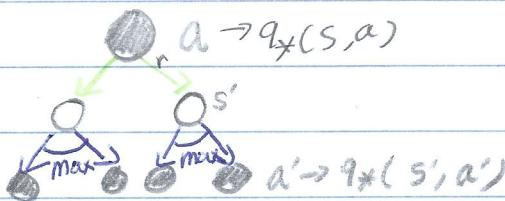
(cf. $q_\pi(s, a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a V_\pi(s')$)



$$V_*(s) = \max_a [R_s^a + \gamma \sum_{s'} P_{ss'}^a V_*(s')]$$

(cf. $V_\pi(s) = \sum_a \pi(a|s) [R_s^a + \gamma \sum_{s'} P_{ss'}^a V_\pi(s')]$)

설명 $V_*(s) = \max_a q_*(s, a)$ 에 $q_*(s, a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a V_*(s')$ 을 대입



$$q_*(s, a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a \max_{a'} q_*(s', a')$$

(cf. $q_\pi(s, a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a \sum_{a'} \pi(a'|s') q_\pi(s', a')$)

설명 $q_*(s, a) = R_s^a + \gamma \sum_{s'} P_{ss'}^a V_*(s')$ 에 $V_*(s) = \max_a q_*(s, a)$ 대입