3. 정책과 가치 함수

순천향대학교 컴퓨터공학과

이 상 정

학습 내용

- 강화학습의 기초 개념
- 정책
- 상태-가치 함수
- 행동-가치 함수
- 최적화된 가치 함수

정책 (Policy)

리턴 (반환값/이득 Return)

- 강화학습의 목적은 **에이전트의 보상을 최대화** 할수 있는 **행동의 집합**을 찾는 일
  - 현재 시점 뿐만 아니라 **미래에 받게 될 보상**
- **리턴 (Return)**은 현재 시점에서 에피소드 종료까지 보상의 총합

Definition

The *return* G_t is the total discounted reward from time-step t.

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

- **할인율** $\gamma \in [0, 1]$ 은 미래의 보상을 현재의 가치로 환산
- k+1 시간 스텝 후의 보상 R의 가치는 $\gamma^k R$.
- 즉시 받는 보상을 미래의 지연된 보상보다 높게 평가
  - 할인율이 0에 가까우면 근시안적인 평가
  - 할인율이 1가까우면 원시적안적인 평가

리턴 (반환값) 표현식

- 리턴의 표현식
  - 보상의 총합

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + ... + R_T$$

  - 할인율 적용

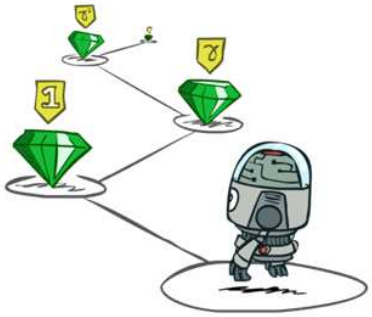
$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... + \gamma^{T-1} R_T$$

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

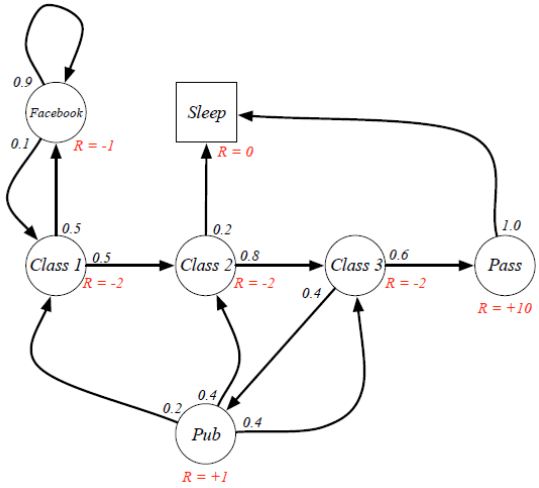
$$G_t = R_{t+1} + \gamma G_{t+1}$$

할인율 적용 이유

- 대부분의 강화학습에서는 다음과 같은 이유로 할인율을 적용
  - 수학적으로 계산이 편리
  - 사이클릭 마르코프 프로세스에서 무한대의 리턴 값을 방지
  - 미래의 불확실성에 대해 할인
  - 보상이 재정적인 경우 즉시 받는 보상이 지연된 보상보다 더 많은 이자 수익 유발
  - 동물/인간의 행동은 즉시 받는 보상을 선호



학생 MRP 리턴 예



Sample returns for Student MRP:
Starting from $S_1 = C1$ with $\gamma = \frac{1}{2}$

$$G_1 = R_2 + \gamma R_3 + ... + \gamma^{T-2} R_T$$

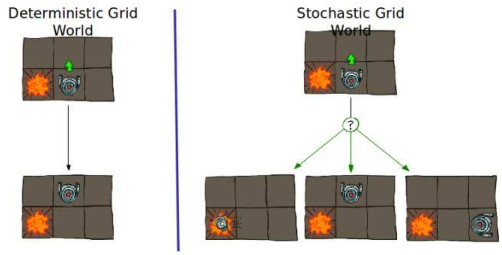
C1 C2 C3 Pass Sleep	$v_1 = -2 - 2 * \frac{1}{2} - 2 * \frac{1}{4} + 10 * \frac{1}{8}$	=	−2.25
C1 FB FB C1 C2 Sleep	$v_1 = -2 - 1 * \frac{1}{2} - 1 * \frac{1}{4} - 2 * \frac{1}{8} - 2 * \frac{1}{16}$	=	−3.125
C1 C2 C3 Pub C2 C3 Pass Sleep	$v_1 = -2 - 2 * \frac{1}{2} - 2 * \frac{1}{4} + 1 * \frac{1}{8} - 2 * \frac{1}{16} ...$	=	−3.41
C1 FB FB C1 C2 C3 Pub C1 ...	$v_1 = -2 - 1 * \frac{1}{2} - 1 * \frac{1}{4} - 2 * \frac{1}{8} - 2 * \frac{1}{16} ...$	=	−3.20
FB FB FB C1 C2 C3 Pub C2 Sleep			

3. 정책과 가치 함수

정책 (Policy)

- **정책 (Policy)**은 특정 상태에서 **에이전트의 행동을 결정**
  - **최적의 정책을 탐색**하는 것이 강화학습의 목표

- **정책은 상태로부터 행동을 매핑**
  - **결정론적 정책 (deterministic policy)**
    - **정형화된 규칙**에 의해 행동을 결정
    - 같은 상태에서는 항상 같은 행동으로 결정
  - **확률론적 정책 (stochastic policy)**
    - **확률적**으로 행동을 결정
    - 같은 상태에서 항상 같은 행동을 결정하지는 않음



MDP 정책

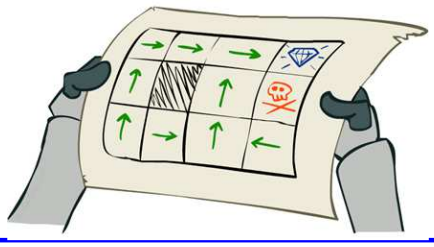
- **MDP 정책 (policy)는 현재 상태에서 에이전트가 어떤 행동 (action)을 취할 확률**

Definition

A *policy* π is a distribution over actions given states,

$$\pi(a|s) = \mathbb{P}[A_t = a \mid S_t = s]$$

- MDP의 정책은 현재의 상태만 고려하고, 과거의 정보는 고려하지 않고 행동
- 확률적으로 행동을 결정
- 정책은 시간 스텝의 변화와 무관하게 독립적



프로즌 레이크 정책 예

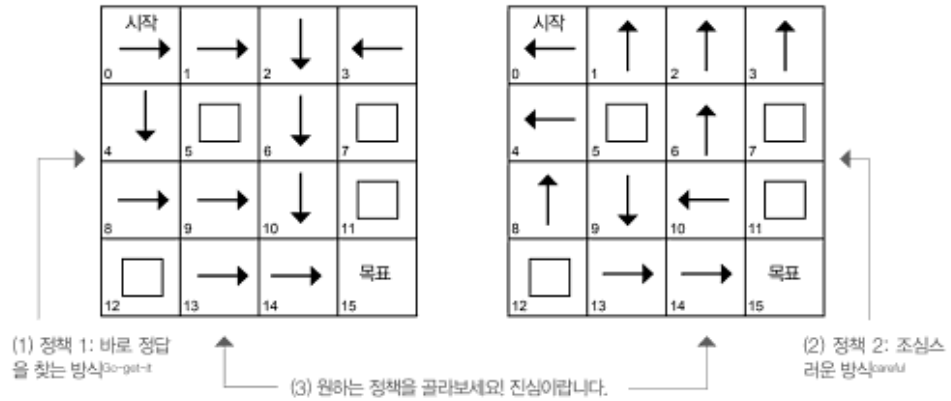


그림 3-8 정책끼리는 어떻게 비교할 수 있을까요?

상태-가치 함수 (Value Function)

가치함수 (Value Function)

- **가치함수 (Value Function)**는 각 **상태와 행동의 가치를 평가**
  - 가치함수는 **미래의 보상을 예측**하여 각 상태 좋음과 나쁨을 평가

- **강화학습**에서는 **가치함수를 정확하게 표현**하는 것이 핵심
  - **미래 가치가 가장 큰 의사결정**을 하고 행동하는 것이 최종 목표

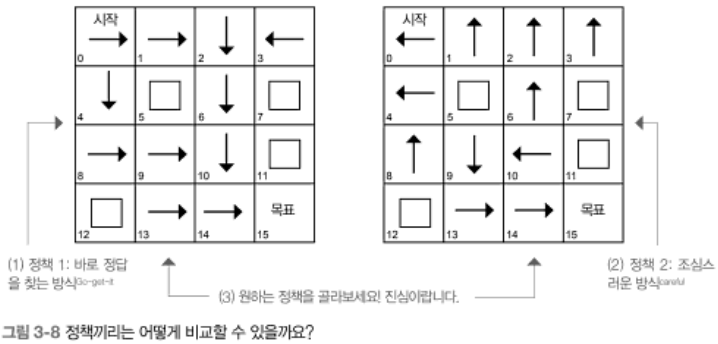


그림 3-8 정책끼리는 어떻게 비교할 수 있을까요?

프로즌 레이크 상태 14 가치 예

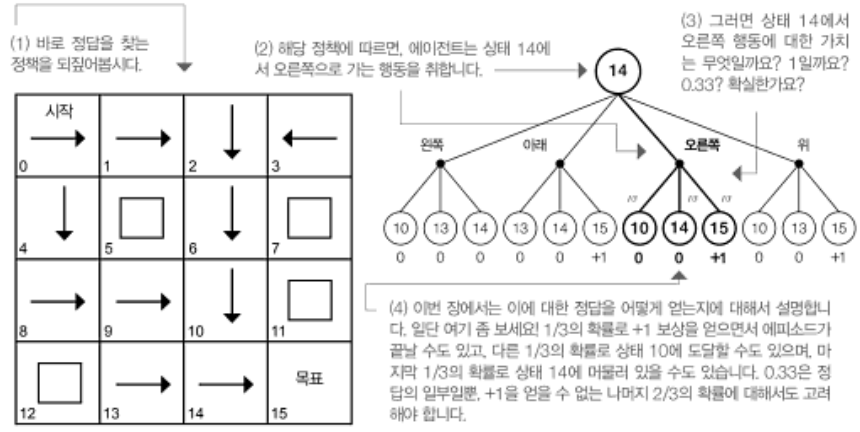


그림 3-9 바로 정답을 찾는 정책을 수행했을 때, 상태 14의 가치는 무엇일까요?

상태-가치함수 (Value Function)

- **상태-가치함수 (state-value function)**는 현재 상태에서 **정책 π** 를 수행할 때 기대되는 미래의 모든 보상의 합(리턴)
  - 현재 상태에서 미래의 모든 기대하는 보상들을 표현
    - 확률적 환경에서 **모든 행동을 고려한 기대값**

Definition

The *state-value function* $v_\pi(s)$ of an MDP is the expected return starting from state s, and then following policy π

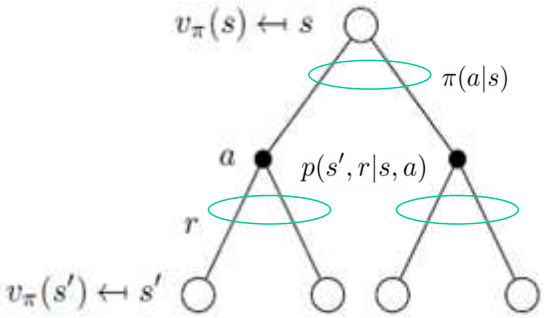
$$v_\pi(s) = \mathbb{E}_\pi [G_t \mid S_t = s]$$

벨만 방정식 (Bellman Equation): 상태-가치 함수

- **상태-가치함수**는 현재 상태에서 정책을 **따르는 즉시 받는 보상**과 할인율을 적용한 **다음 상태의 가치**로 분리 표현
  - 이 방정식을 **벨만 방정식(bellman Equation)**이라고 함
  - **즉시 받는 보상** R_{t+1}
  - **할인된 다음 상태의 가치함수** $\gamma v(S_{t+1})$

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

$$v_\pi(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s \right]$$



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

상태-가치 함수 표현식

수식으로 이해하기: 상태-가치 함수 V

(1) 상태 s에 대한 가치

(3) π에 대한 기대치를 나타냅니다.

$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s]$$

(5) 타임 스텝 t에서 상태 s에 있을 때

(2) 정책 π를 수행하고 있을 때

(4) 타임 스텝 t에서의 반환값

(6) 반환값은 감가된 보상들의 총합이라는 것을 기억하세요.

$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots|S_t = s]$$

(7) 그리고 이와 같이 재귀의 형태로 정의할 수 있습니다.

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

(8) 이 공식을 벨만 방정식bellman equation이라고 하고, 상태들에 대한 가치를 찾을 수 있도록 해줍니다.

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

(9) 여기서 상태 s에서 수행해야 할 행동(정책이 확률적일 경우, 행동이 여러 개 나올 수 있습니다)을 얻을 수 있습니다. 여기에 가중치를 가한 값들의 합을 구합니다.

(10) 또한 다음 상태와 보상의 확률에 대해서도 가중치를 가하고 이에 대한 합을 구할 수 있습니다.

(11) 보상과 도착한 상태에서의 감가된 가치를 더하고, 발생할 수 있는 전이에 대한 확률을 가중치로 가합니다.

(12) 상태 영역상의 모든 상태에 대해서 수행합니다.

행동-가치 함수 (Action-Value Function)

행동-가치함수 (Action-Value Function)

- 상태와 행동까지 모두 고려하는 경우는 **행동-가치함수 (action-value Function)** 이라고 함
  - **Q-함수** 라고도 함
  - 서로 다른 행동을 비교하여 좋은 행동을 선택하여 정책을 개선

- **행동-가치함수 (action-value function) q**은 현재 상태에서 **특정 행동을 취하는 조건에서 정책**의 기대되는 미래의 모든 보상의 합(리턴)
  - 현재 상태에서 **특정 행동만 고려**한 가치

Definition

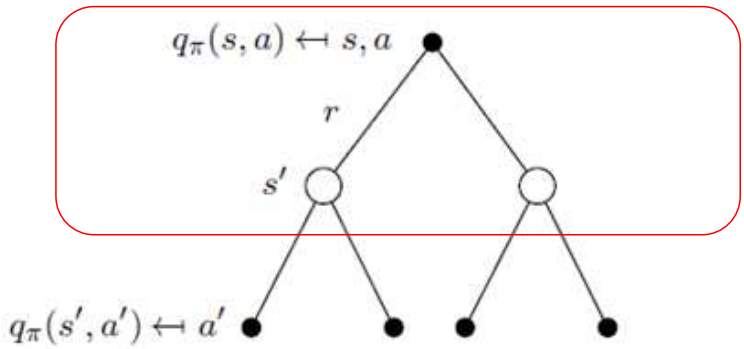
The *action-value function* $q_\pi(s, a)$ is the expected return starting from state s, taking action a, and then following policy π

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

벨만 방정식: 행동-가치 함수 (Bellman Expectation Equation)

- **행동-가치함수**는 현재 상태에서 정책에 따른 취하여 전이되는 **모든 다음 상태-가치 함수의 합**

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')], \forall s \in S, \forall a \in A(s)$$



행동-가치 함수 표현식

수식으로 이해하기: 행동-가치 함수 Q

(1) 정책 π를 따르는 동안, 상태 s에서 행동 a를 취했을 때에 대한 가치

(2) 이는 결과적으로 정책 π를 따르는 동안, 상태 s에서 행동 a를 취했을 때의 반환값에 대한 기대치를 말합니다.

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

(3) 이와 같이 함수를 재귀적으로 정의할 수 있습니다.

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

(4) 행동 가치에 대한 벨만 방정식은 다음과 같이 정의할 수 있습니다.

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')], \forall s \in S, \forall a \in A(s)$$

(5) 우리는 특정 행동에 대해서만 신경쓰기 때문에 행동에 대해서는 가중치를 줄 필요가 없습니다.

(6) 여기서 가중치를 주기는 하지만 이는 다음 상태와 보상에 대한 확률에 대한 가중치 입니다.

(7) 여기서 어디에 가중치를 줄까요? 바로 보상과 다음 상태에서의 감가된 가치의 합에 줄 수 있습니다.

(8) 환경에서 얻은 모든 상태-행동 쌍에 적용할 수 있습니다.

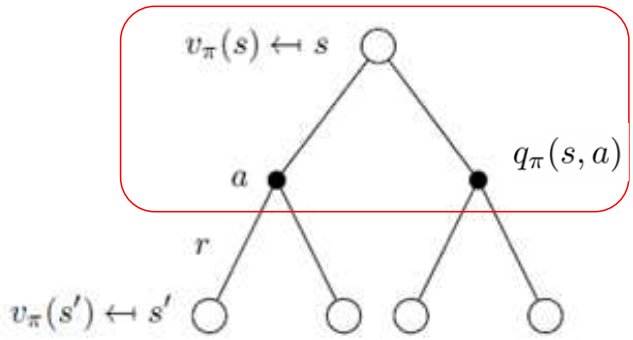
벨만 방정식

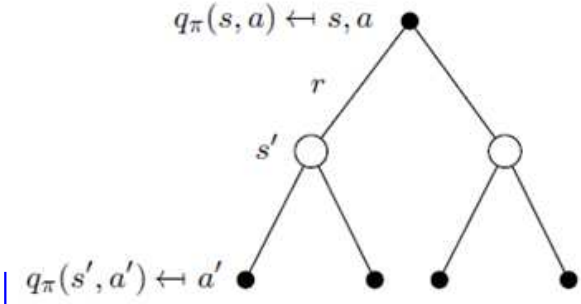
- 상태-가치 함수는 행동-가치 함수의 합으로 표현

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

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

$$q_\pi(s, a) = \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')], \forall s \in S, \forall a \in A(s)$$





최적화된 가치 함수
(Optimal Value Function)

최적화된 가치함수 (Optimal Value Function)

❑ 가치함수의 최적화

- MDP의 정책 중에서 **최대의 가치를 갖는 정책**의 가치함수
  - 최적화된 상태-가치함수, 최적화된 행동-가치함수
- **MDP의 해(solution)**은 최적화된 가치함수를 구하는 것

Definition

The *optimal state-value function* $v_*(s)$ is the maximum value function over all policies

$$v_*(s) = \max_\pi v_\pi(s)$$

The *optimal action-value function* $q_*(s,a)$ is the maximum action-value function over all policies

$$q_*(s,a) = \max_\pi q_\pi(s,a)$$

최적화된 정책 발견

❑ 최적화된 정책은 각 상태에서 최적화된 행동-가치함수 q_*를 최대화하는 것

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \underset{a \in \mathcal{A}}{\text{argmax}}\, q_*(s,a) \\ 0 & \text{otherwise} \end{cases}$$

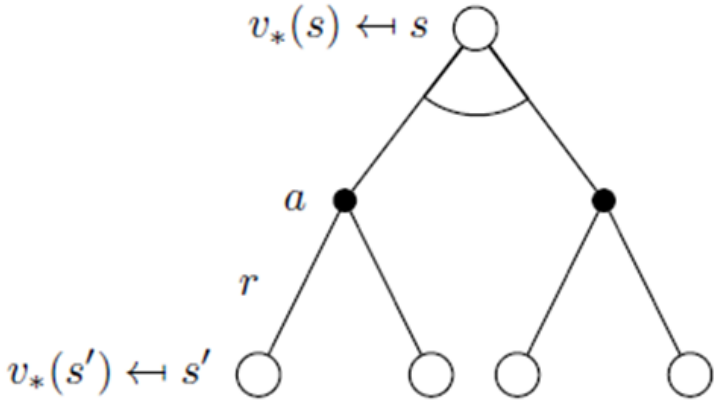
- q_*가 최대값이되는 행동을 선택하면 되므로, q*(s,a)를 알면 최적화된 정책을 알 수 있음

벨만 최적화 방정식: 최적화된 상태-가치함수 V*

$$v_*(s) = \max_\pi v_\pi(s), \forall s \in S$$

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

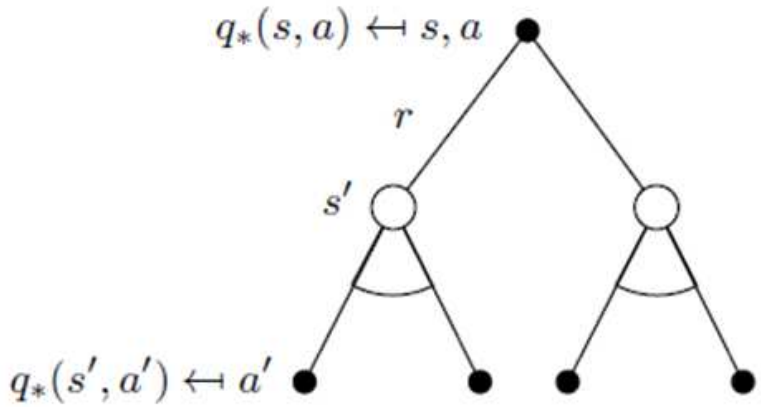
$$q_\pi(s, a)$$



벨만 최적화 방정식: 최적화된 행동-가치함수

$$q_*(s, a) = \max_\pi q_\pi(s, a), \forall s \in S, \forall a \in A(s)$$

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



최적화된 가치 함수 표현식

(1) 이상적인 상태-가치 함수 → $v_*(s) = \max_\pi v_\pi(s), \forall s \in S$ ← (2) 모든 정책에 대해 가장 높은 가치를 얻는 상태-가치 함수입니다.

(3) 동일하게, 이상적인 행동-가치 함수는 가장 높은 가치를 얻는 행동-가치 함수입니다. → $q_*(s,a) = \max_\pi q_\pi(s,a), \forall s \in S, \forall a \in A(s)$

(4) 이상적인 상태-가치 함수는 다음과 같이 구할 수 있습니다. → $v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')]$

(5) 여기서 최댓값 연산 max를 취할 수 있습니다. →

(6) 이 부분은 보상과 다음 상태에서의 감가된 이상치와의 합에 대해 가중치가 더해진 것입니다. →

(7) 이와 유사하게, 이상적인 행동-가치 함수는 이와 같이 구할 수 있습니다. → $q_*(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma \max_{a'} q_*(s',a')\right]$

(8) 여기에 max 연산이 안에 들어가 있는 것을 확인할 수 있습니다. →

벨만 최적화 방정식 해

- 벨만 최적화 방정식에서 최적화된 가치 함수를 찾으면 최적화된 정책도 구할 수 있음
  - 벨만 최적화 방정식은 **비선형 함수 이므로** 일반적인 해는 없음
- 일반적으로 큰 MRP는 아래와 같은 **반복적인 방식 (iterative method)** 사용하여 해를 구함
  - **동적 계획법 (Dynamic Programming)**
  - **강화학습 (Reinforcement Learning)**

참고 자료

- David Silver - UCL Course on RL, 2015
  - https://www.davidsilver.uk/teaching/
  - Lecture 2: Markov Decision Processes
- Miguel Morales, Grokking Deep Reinforcement Learning
  - https://livebook.manning.com/book/grokking-deep-reinforcement-learning
  - 그로킹 심층 강화학습, 강찬석 옮김, 한빛미디어
  - 3장 목표와 장기 목표 간의 균형