Bellman Equation

Slides from

1. 이웅원 외, 파이썬과 케라스로 배우는 강화학습, 주교재
2. 이웅원, 가깝고도 먼 DeepRL, PPT
3. David Silver, Reinforcement Learning, PPT

References

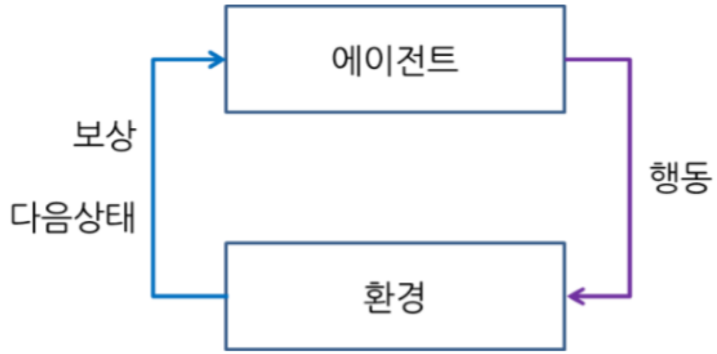
1. Richard S. Sutton and Andrew G. Barto, Reinforcement Learning: An Introduction, MIT Press
2. 유튜브, 전민영, 노승은, 강화학습의 기초 이론, 팡요랩

Contents

MDP 에이전트의 행동 선택

1. 에이전트와 환경의 상호작용
   (1) 에이전트가 환경에서 자신의 상태를 관찰
   (2) 그 상태에서 어떠한 기준에 따라 행동을 선택
   - 어떠한 기준 : 가치함수,
   - 행동 선택 : 정책

   (3) 선택한 행동을 환경에서 실행
   (4) 환경으로부터 다음상태와 보상을 받음
   (5) 보상을 통해 에이전트가 가진 정보(가치함수)를 수정함
   (6) 위의 과정을 반복하여 최적의 정책을 학습



$$s_0, a_0, r_1, s_1, a_1, r_2, \cdots, s_T$$

MDP 에이전트의 행동 선택

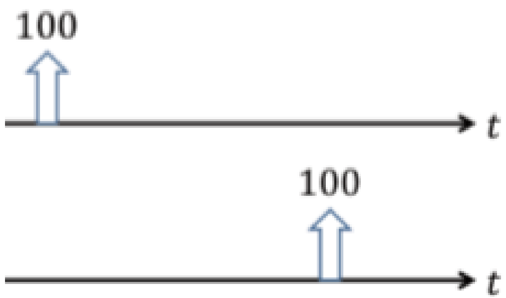
2. 에이전트 행동 선택의 기준

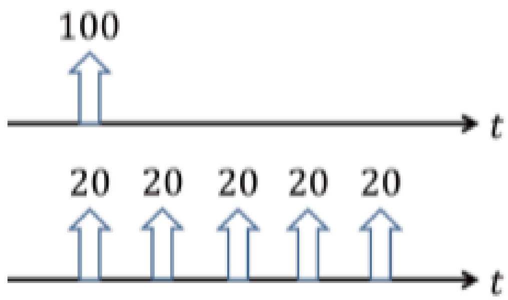
- 앞으로 받을 보상의 합을 고려해서 선택
- 아직 받지 않은 보상들을 어떻게 고려? => 가치함수 (Value Function)

보상의 표현 1 : 단순합

- 현재 시간 t로부터 앞으로 받을 보상을 다 더한다

$$R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$$

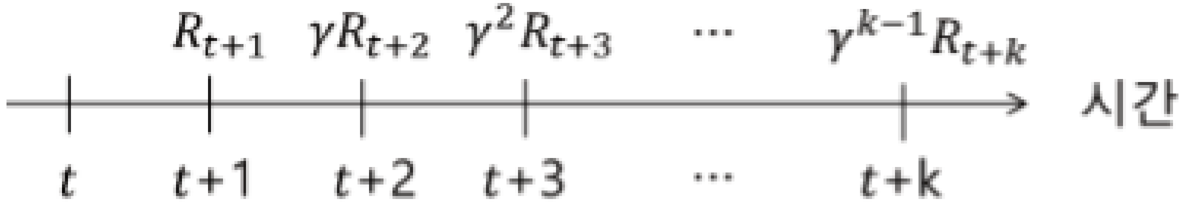




$$0.1 + 0.1 + \cdots = \infty$$
$$1 + 1 + \cdots = \infty$$

보상의 표현 2 : 반환값 (Return)

- 현재 시간 t로부터 에피소드 끝까지 받은 보상을 할인해서 현재 가치로



- 반환값(Return) : 현재 가치로 변환한 보상들을 다 더한 값

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_{t+T}$$

Return

Definition

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

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

- The *discount* $\gamma \in [0, 1]$ is the present value of future rewards
- The value of receiving reward R after $k + 1$ time-steps is $\gamma^k R$.
- This values immediate reward above delayed reward.
    - γ close to 0 leads to "myopic" evaluation
    - γ close to 1 leads to "far-sighted" evaluation

David Silver, Reinforcement Learning, PPT

Why discount?

Most Markov reward and decision processes are discounted. Why?

- Mathematically convenient to discount rewards
- Avoids infinite returns in cyclic Markov processes
- Uncertainty about the future may not be fully represented
- If the reward is financial, immediate rewards may earn more interest than delayed rewards
- Animal/human behaviour shows preference for immediate reward
- It is sometimes possible to use *undiscounted* Markov reward processes (i.e. $\gamma = 1$), e.g. if all sequences terminate.

David Silver, Reinforcement Learning, PPT

보상의 표현 3 : 가치함수

- 가치함수(Value function) : 반환값에 대한 기댓값
  - 어떠한 상태 s에 갈 경우 그 이후로 받을 것이라 예상되는 보상에 대한 기대값

$$v(s) = \boldsymbol{E}[G_t | S_t = s] \qquad G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_{t+T}$$

- 현재 에이전트가 갈 수 있는 상태들의 가치를 안다면 그 중에서 가장 가치가 높은 상태를 선택할 수 있음
- 기댓값을 계산하기 위해서는 환경의 모델을 알아야함
  - Dynamic Programming은 가치함수를 계산
  - 강화학습은 가치함수를 계산하지 않고 sampling을 통한 approximation

보상의 표현 3 : 가치함수

- 반환값(Return)의 식을 이용해서 $v(s)$를 다시 쓰면

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_{t+T}$$

$$v(s) = \boldsymbol{E}[G_t | S_t = s]$$

$$\Rightarrow$$

$$v(s) = \boldsymbol{E}[R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_{t+T} \,|S_t = s]$$

$$v(s) = \boldsymbol{E}[R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \cdots) \,|S_t = s]$$

$$v(s) = \boldsymbol{E}[R_{t+1} + \gamma G_{t+1} \,|S_t = s]$$

$$v(s) = \boldsymbol{E}[R_{t+1} + \gamma v(S_{t+1}) \,|S_t = s]$$

Markov Reward Process

A Markov reward process is a Markov chain with values.

Definition

A *Markov Reward Process* is a tuple $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ is a finite set of states
- $\mathcal{P}$ is a state transition probability matrix,
  $\mathcal{P}_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s]$
- $\mathcal{R}$ is a reward function, $\mathcal{R}_s = \mathbb{E}[R_{t+1} \mid S_t = s]$
- γ is a discount factor, $\gamma \in [0, 1]$

David Silver, Reinforcement Learning, PPT

Value function of MRP

The value function $v(s)$ gives the long-term value of state s

Definition

The *state value function* $v(s)$ of an MRP is the expected return starting from state s

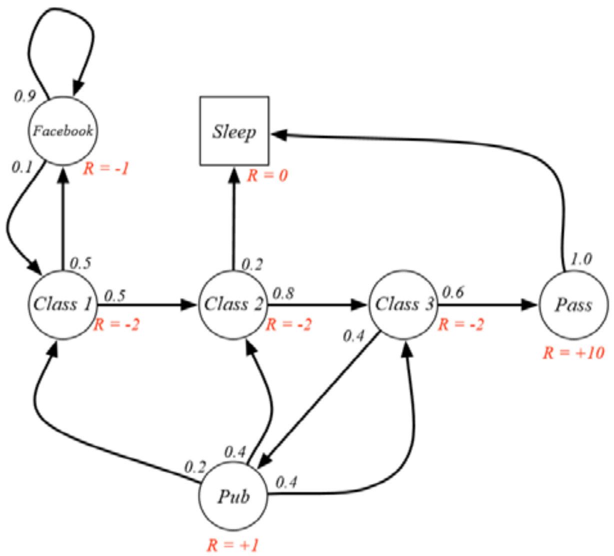
$$v(s) = \mathbb{E}\left[G_t \mid S_t = s\right]$$

David Silver, Reinforcement Learning, PPT

Example: Student MRP Returns

Sample **returns** for Student MRP:

Starting from $S_1 = \text{C1}$ with $\gamma = \frac{1}{2}$

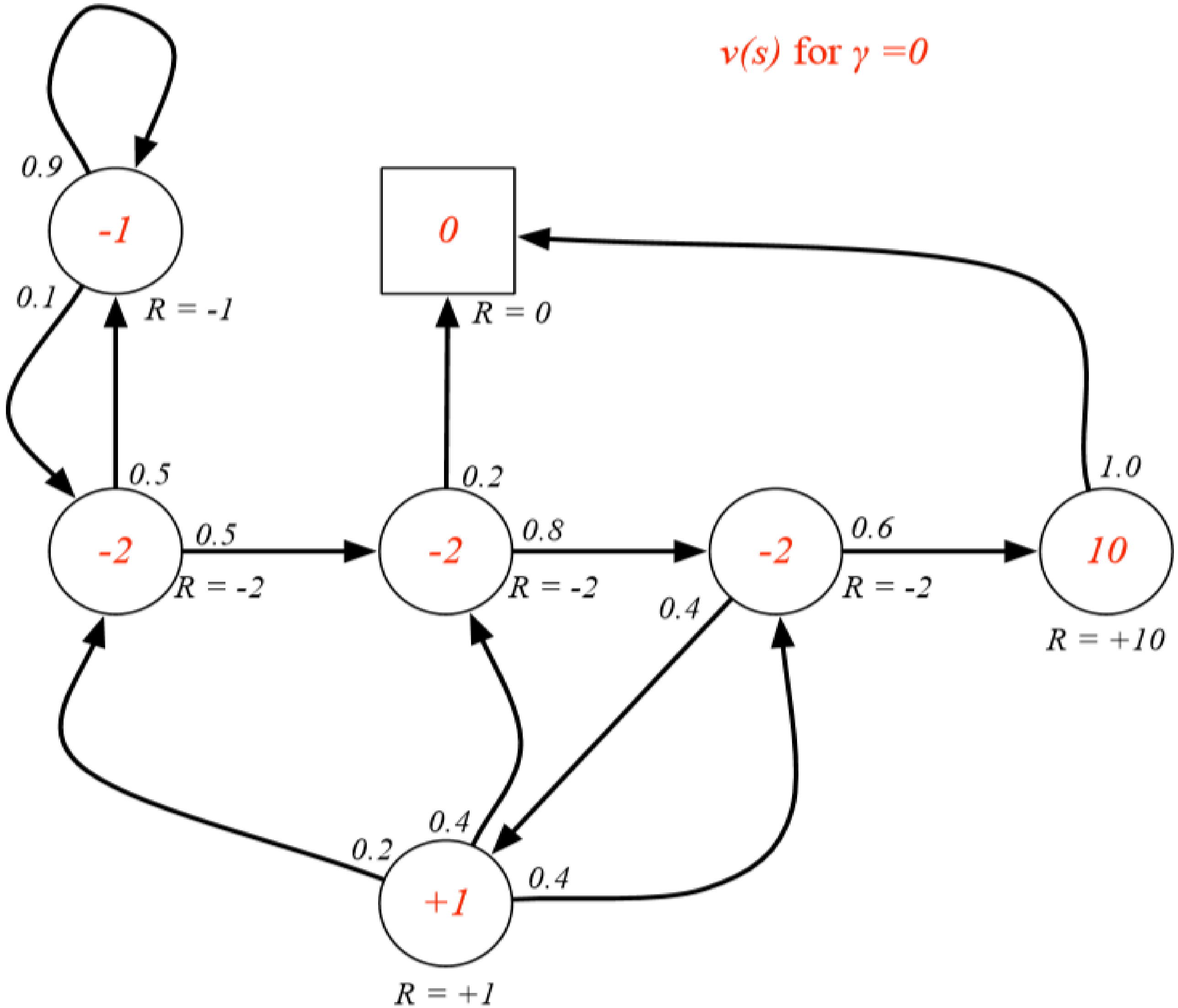


$$G_1 = R_2 + \gamma R_3 + ... + \gamma^{T-2} R_T$$

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

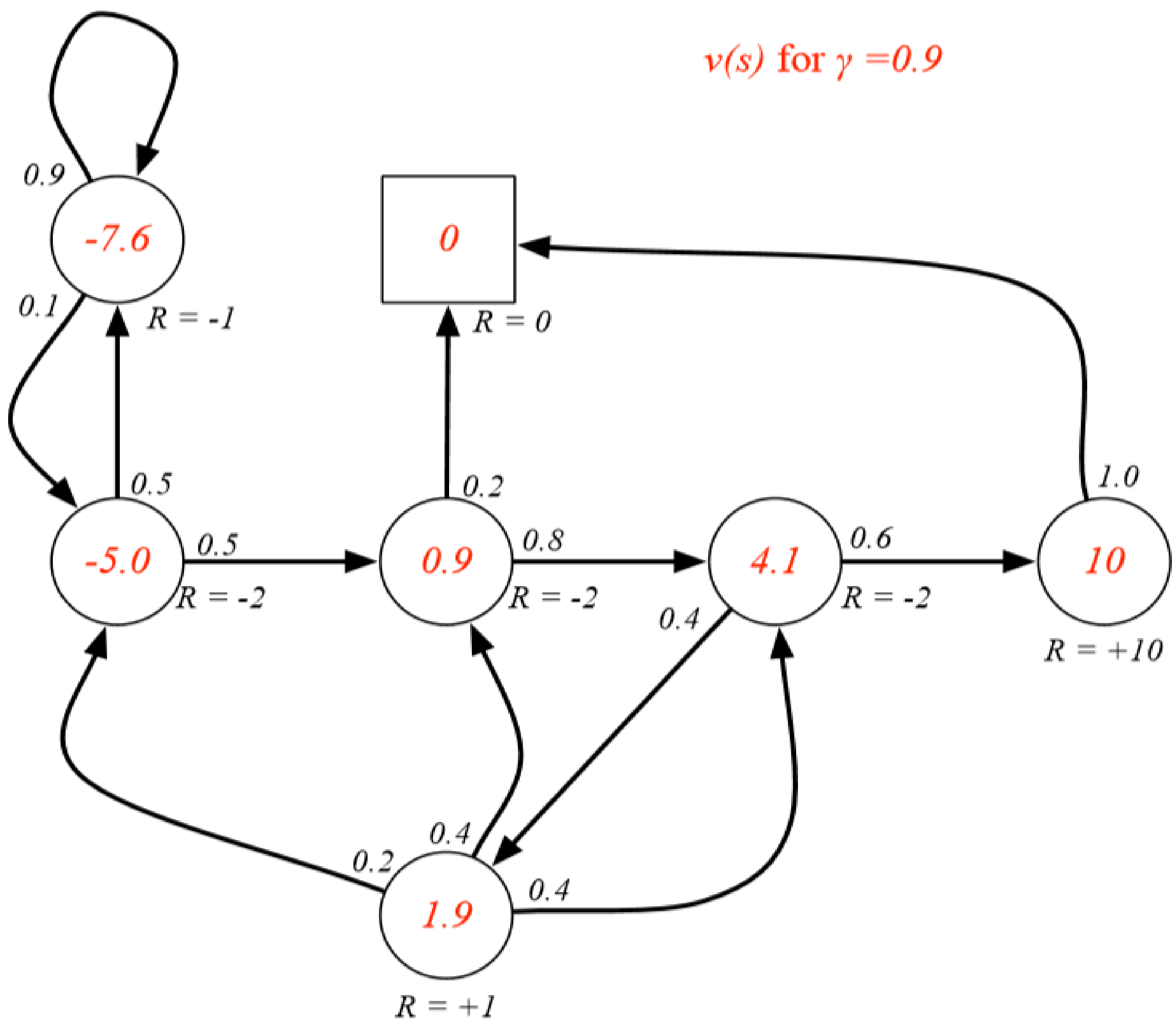
David Silver, Reinforcement Learning, PPT

Example: State-Value Function for Student MRP (1)



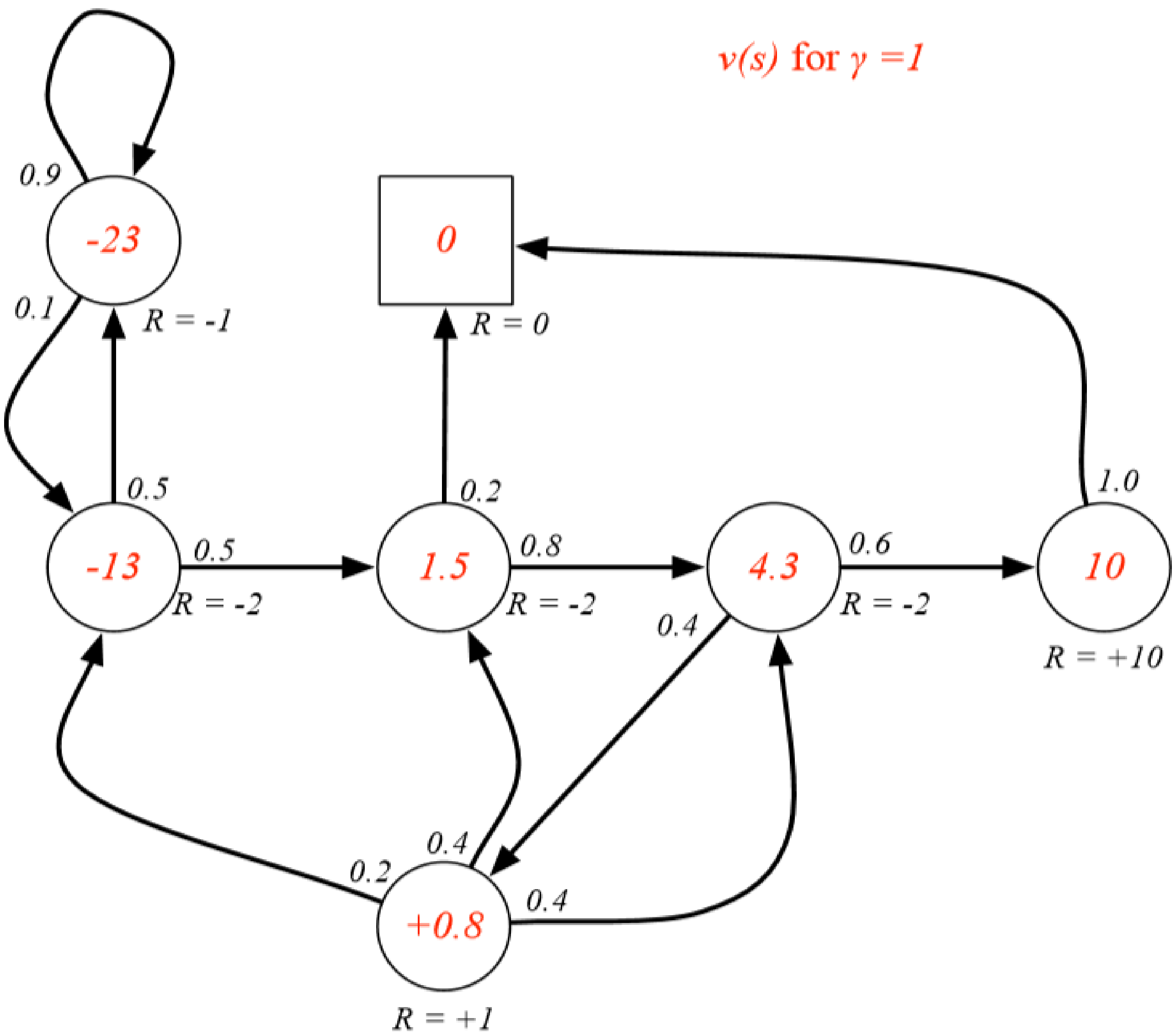
David Silver, Reinforcement Learning, PPT

Example: State-Value Function for Student MRP (2)



David Silver, Reinforcement Learning, PPT

Example: State-Value Function for Student MRP (3)



David Silver, Reinforcement Learning, PPT

Bellman Equation of MRP

The value function can be decomposed into two parts:

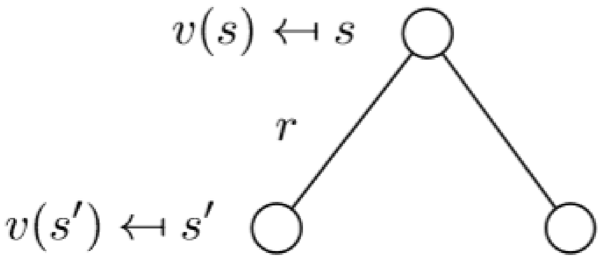
- immediate reward R_{t+1}
- discounted value of successor state $\gamma v(S_{t+1})$

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

David Silver, Reinforcement Learning, PPT

Bellman Equation of MRP (2)

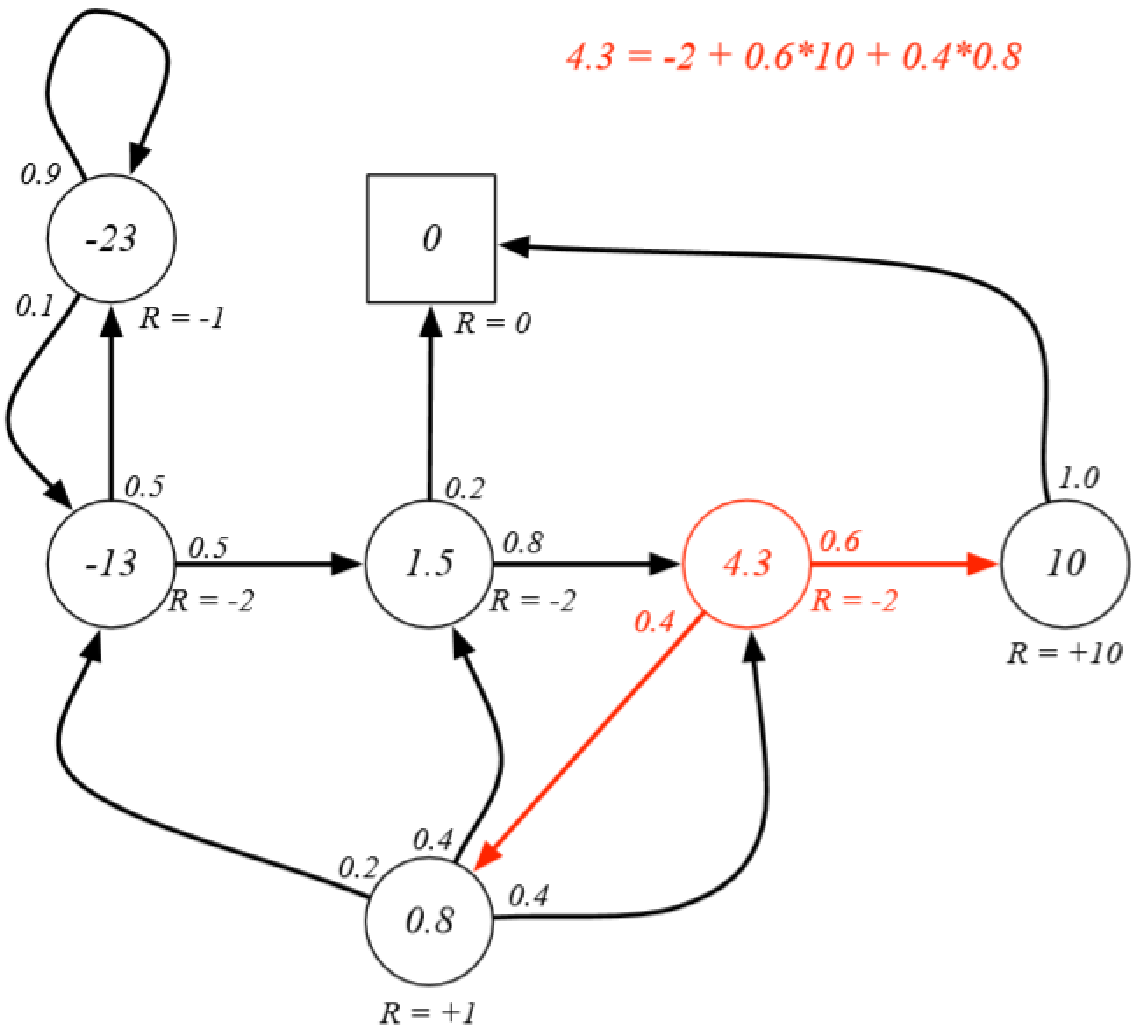
$$v(s) = \mathbb{E}\left[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s\right]$$



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

David Silver, Reinforcement Learning, PPT

Example: Bellman Equation for Student MRP



David Silver, Reinforcement Learning, PPT

Markov Decision Process

A Markov decision process (MDP) is a Markov reward process with decisions. It is an *environment* in which all states are Markov.

Definition

A *Markov Decision Process* is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ is a finite set of states
- $\mathcal{A}$ is a finite set of actions
- $\mathcal{P}$ is a state transition probability matrix,
  $\mathcal{P}_{ss'}^{a} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a\right]$
- $\mathcal{R}$ is a reward function, $\mathcal{R}_s^a = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$
- γ is a discount factor $\gamma \in [0, 1]$.

David Silver, Reinforcement Learning, PPT

Policies (1)

Definition

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

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

- A policy fully defines the behaviour of an agent
- MDP policies depend on the current state (not the history)
- i.e. Policies are *stationary* (time-independent),
  $A_t \sim \pi(\cdot|S_t), \forall t > 0$

David Silver, Reinforcement Learning, PPT

정책을 고려한 가치함수

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

$$v_\pi(s) = \boldsymbol{E}_\pi[R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_{t+T} | S_t = s]$$

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

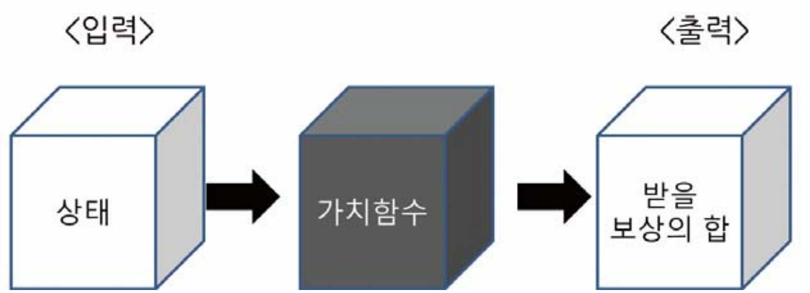
$$v_\pi(s) = \boldsymbol{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s]$$

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

=> 벨만 기대 방정식 (Bellman Expectation Equation)

(상태)가치함수 (State Value Function)

$$v_\pi(s) = \boldsymbol{E}_\pi[G_t|S_t = s] \ldots = \boldsymbol{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})\,|S_t = s]$$

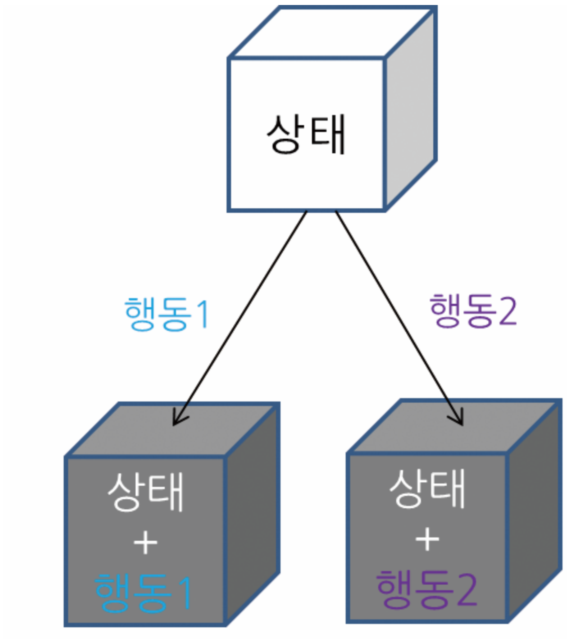


- 상태가 입력으로 들어오면 그 상태에서 앞으로 받을 보상의 합을 출력 -> 에이전트가 어떤 상태에 있는 것이 얼마나 좋은지를 알 수 있음

행동가치함수 (Action Value Function)

- 상태 s에서 행동 a를 했을 경우 받을 것이라 예상되는 반환값에 대한 기댓값
- 어떤 상태에서 어떤 행동을 한 후의 가치함수
- 어떤 상태에서 어떤 행동이 얼마나 좋은지를 알려주는 함수
- aka 큐함수(Q Function)

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



큐함수에 대한 벨만 방정식

$$q_\pi(s,a) = \boldsymbol{E_\pi}[G_t|S_t = s, A_t = a]$$

$$q_\pi(s,a) = \boldsymbol{E_\pi}[R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_{t+T}\,|S_t = s, A_t = a]$$

$$q_\pi(s,a) = \boldsymbol{E_\pi}[R_{t+1} + \gamma G_{t+1}|S_t = s, A_t = a]$$

$$q_\pi(s,a) = \boldsymbol{E_\pi}[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1})|S_t = s, A_t = a]$$

=> 벨만 기대 방정식 (Bellman Expectation Equation)

가치함수 vs 큐함수

- 상태가치함수는 큐함수에 대한 기댓값

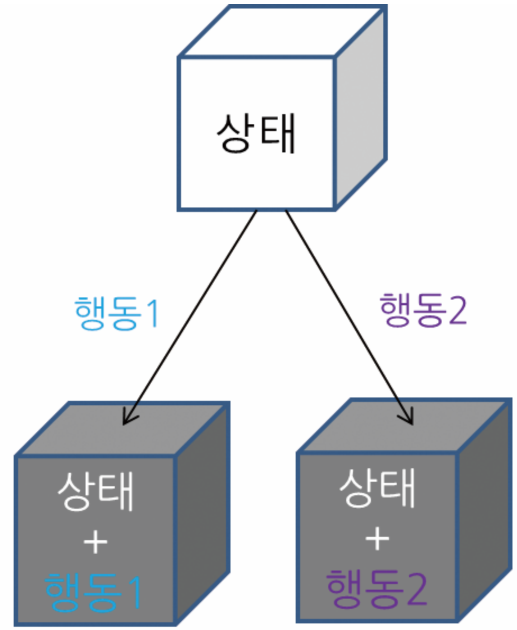
$$v_\pi(s) = \boldsymbol{E}_{a\sim\pi}[q_\pi(s,a)|S_t = s]$$

$$v_\pi(s) = \sum_{a\in A} \pi(a|s)q_\pi(s,a)$$

cf)

$$q_\pi(s,a) = \boldsymbol{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s, A_t = a]$$

$$q_\pi(s,a) = R_s^a + \gamma v_\pi(s')$$



Value functions of MDP

Definition

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

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

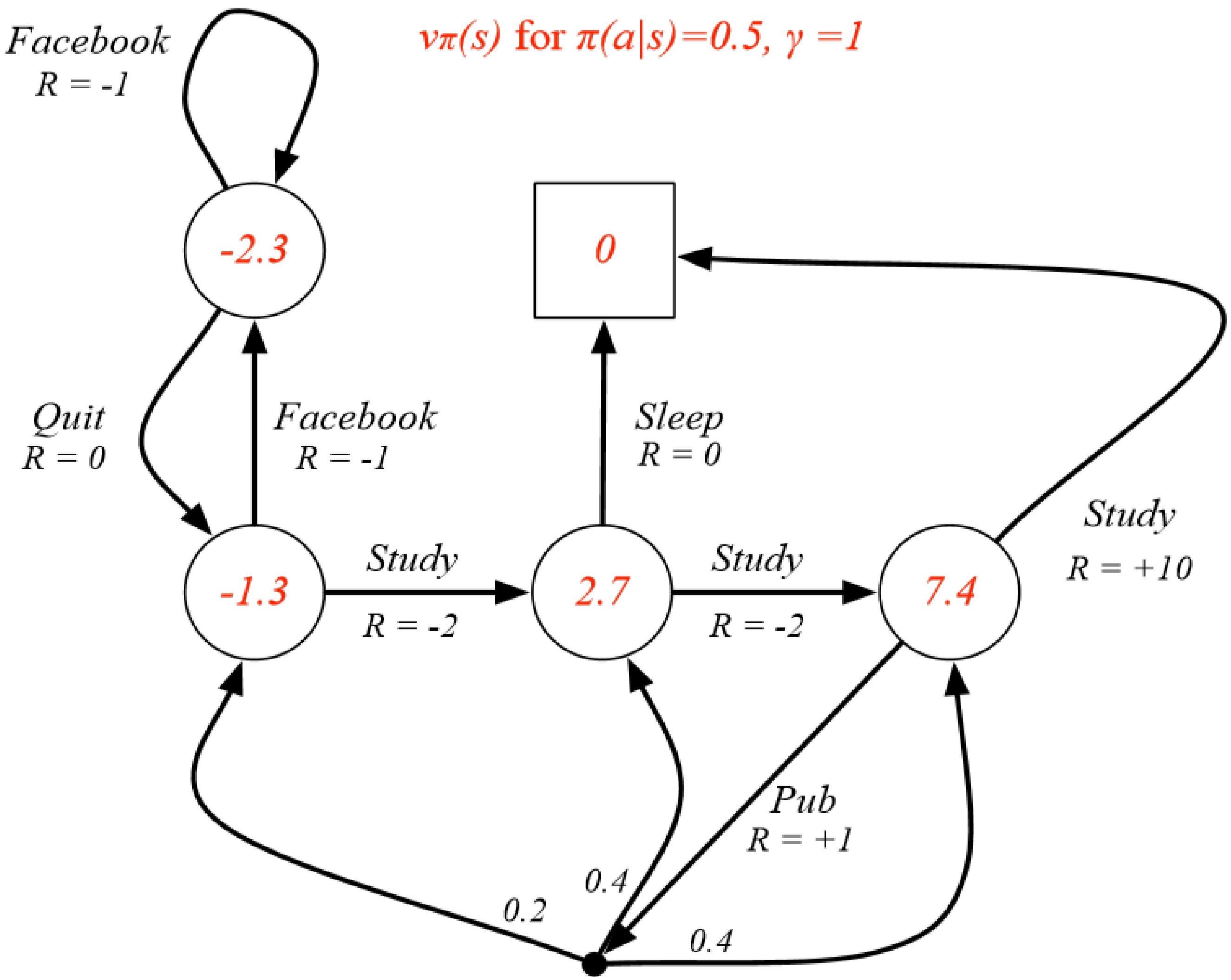
Definition

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

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

David Silver, Reinforcement Learning, PPT

Example: State-Value Function for Student MDP



David Silver, Reinforcement Learning, PPT

Bellman Expectation Equation

The state-value function can again be decomposed into immediate reward plus discounted value of successor state,

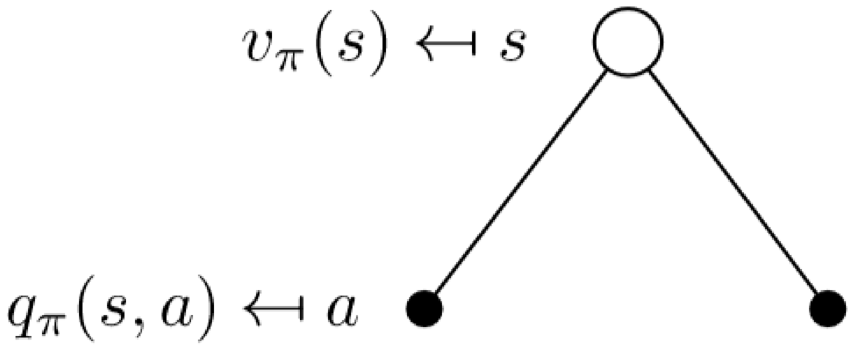
$$v_\pi(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s \right]$$

The action-value function can similarly be decomposed,

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

David Silver, Reinforcement Learning, PPT

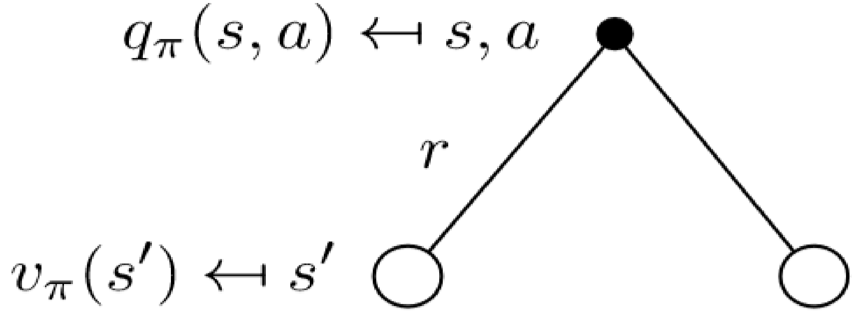
Bellman Expectation Equation for V^{π}



$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) q_\pi(s,a)$$

David Silver, Reinforcement Learning, PPT

Bellman Expectation Equation for Q^{π}



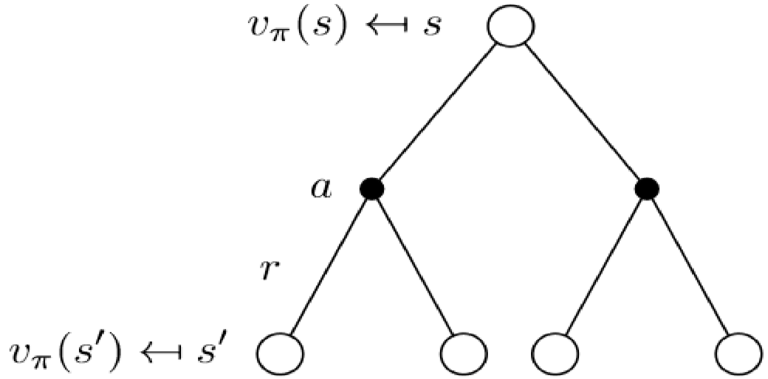
$$q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s')$$

if $P_{ss'}^{a} = 1$, $q_\pi(s,a) = R_s^a + \gamma v_\pi(s')$

-> $q_\pi(s,a) = \boldsymbol{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s, A_t = a]$

David Silver, Reinforcement Learning, PPT

Bellman Expectation Equation for v^{π} (2)

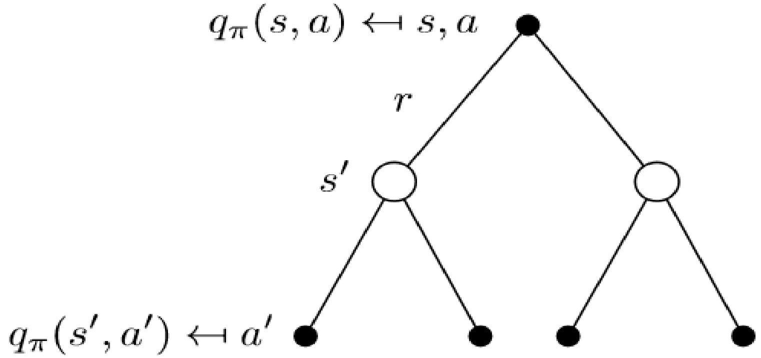


$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s') \right)$$

if $P_{ss'}^{a} = 1$, $v_\pi(s) = \sum_{a \in A} \pi(a|s)(R_s^a + \gamma v_\pi(s'))$

David Silver, Reinforcement Learning, PPT

Bellman Expectation Equation for q^{π} (2)



$$q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \sum_{a' \in \mathcal{A}} \pi(a'|s') q_\pi(s',a')$$

if $P_{ss'}^{a} = 1$, $q_\pi(s,a) = R_s^a + \gamma \sum_{a \in A} \pi(a'|s')(q_\pi(s',a'))$

David Silver, Reinforcement Learning, PPT

벨만 기대 방정식 계산

- 벨만 방정식은 현재 상태 s와 다음 상태 S_{t+1}의 가치함수/큐함수 사이의 관계식

-> 벨만 기대 방정식을 반복적으로 계산하면 참 가치함수를 구할 수 있음

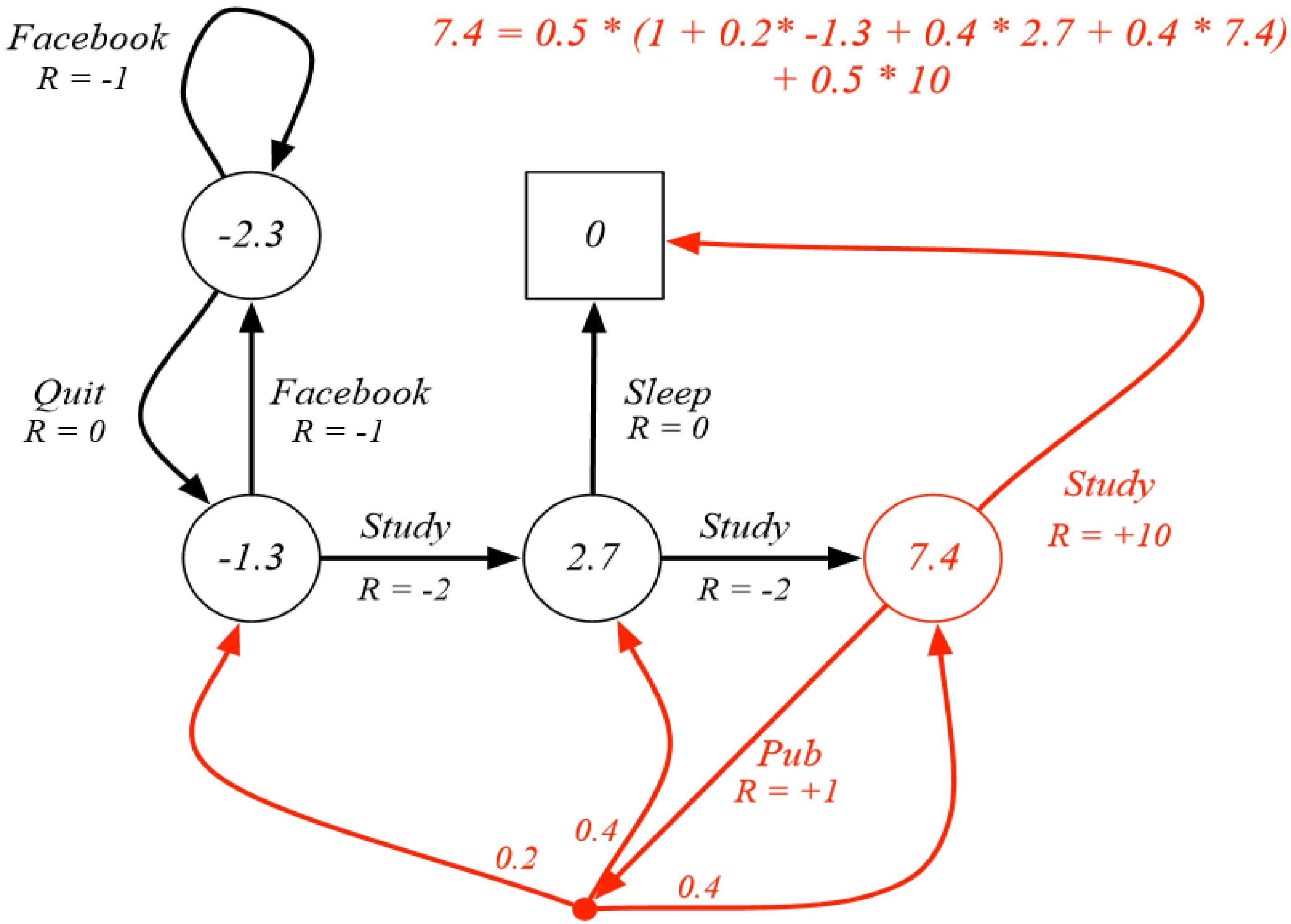
$$v_\pi(s) = \boldsymbol{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s]$$

$$=> v_{k+1}(s) = \sum_{a \in A} \pi(a|s)(R_s^a + \gamma v_k(s'))$$

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

$$=> q_{k+1(}s,a) = R_s^a + \gamma \sum_{a \in A} \pi(a'|s')(q_k(s',a'))$$

Example: Bellman Expectation Equation in Student MDP

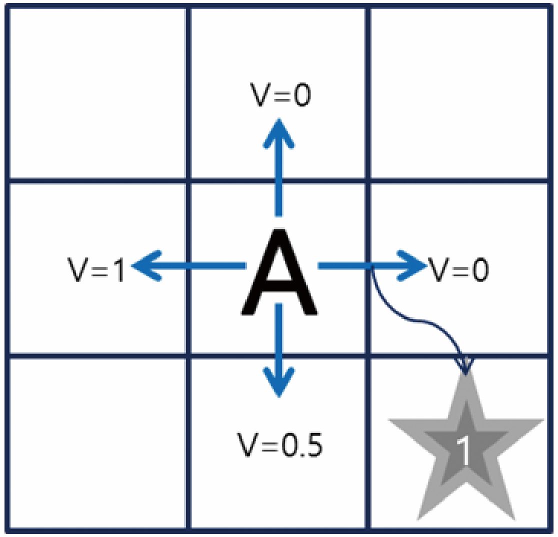


David Silver, Reinforcement Learning, PPT

Grid world에서 벨만 기대 방정식 계산

$$v_\pi(s) = \sum_{a\in A} \pi(a|s)(R_s^a + \gamma v_\pi(s'))$$

Assume $P_{ss'}^a = 1$



상: $0.25 \times (0 + 0.9 \times 0) = 0$

하: $0.25 \times (0 + 0.9 \times 0.5) = 0.1125$

좌: $0.25 \times (0 + 0.9 \times 1) = 0.225$

우: $0.25 \times (1 + 0.9 \times 0) = 0.25$

다음 가치함수 = 0 + 0.1125 + 0.225 + 0.25

$\pi(상|s) = \pi(하|s) = \pi(좌|s) = \pi(우|s) = 0.25,\ \gamma = 0.9$

Bellman Expectation Equation (Matrix Form)

The Bellman expectation equation can be expressed concisely using the induced MRP,

$$v_\pi = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi v_\pi$$

with direct solution

$$v_\pi = \left(I - \gamma \mathcal{P}^\pi \right)^{-1} \mathcal{R}^\pi$$

David Silver, Reinforcement Learning, PPT

벨만 기대 방정식과 최적의 정책

- 벨만 기대 방정식 -> 정책 π를 따라갔을 때의 가치함수
  -> 참 가치함수

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

- 에이전트가 알고자 하는 것 : π^*
  - 가장 높은 보상을 얻게 하는 최적의 정책

- 최적의 정책이란?
  - 가장 큰 가치함수, 즉 최적 가치함수를 주는 정책

$$v^*(s) = \max_\pi[v_\pi(s)]$$

벨만 기대 방정식과 최적의 정책

- 최적의 큐함수 -> 정책이 최적일 때 그에 따르는 큐함수도 최적

$$q^*(s,a) = \max_\pi[q_\pi(s)]$$

- 최적의 정책 (optimal policy)

$$\pi^*(a|s) = \underbrace{\mathrm{argmax}_{a \in A}}[q^*(s,a)]$$

Greedy policy : 가능한 행동 중에서 최고의 큐함수를 가지는 행동을 선택하는 정책

Optimal Value Function

Definition

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

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

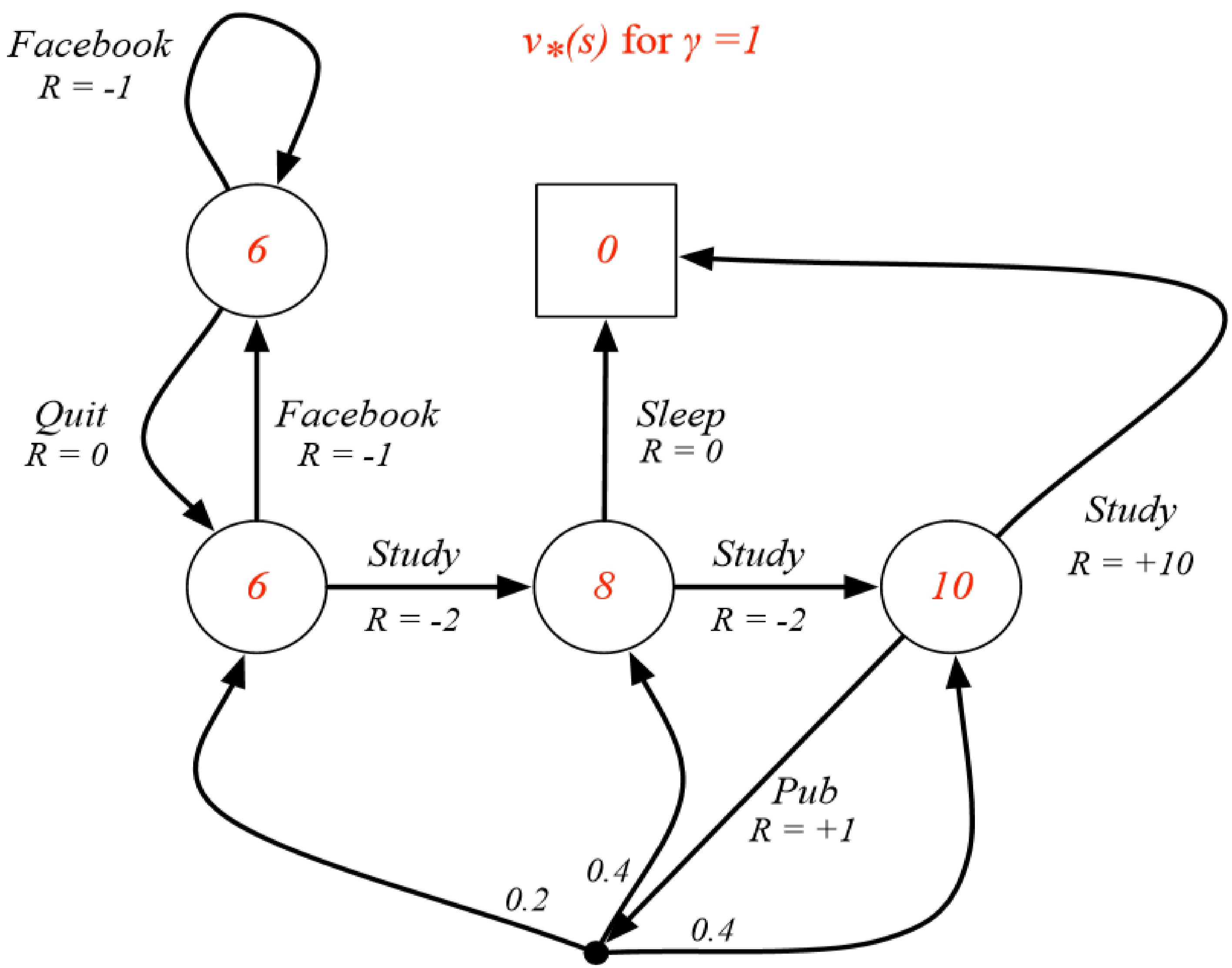
The *optimal action-value function* $q_*(s, a)$ is the maximum action-value function over all policies

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

- The optimal value function specifies the best possible performance in the MDP.
- An MDP is "solved" when we know the optimal value fn.

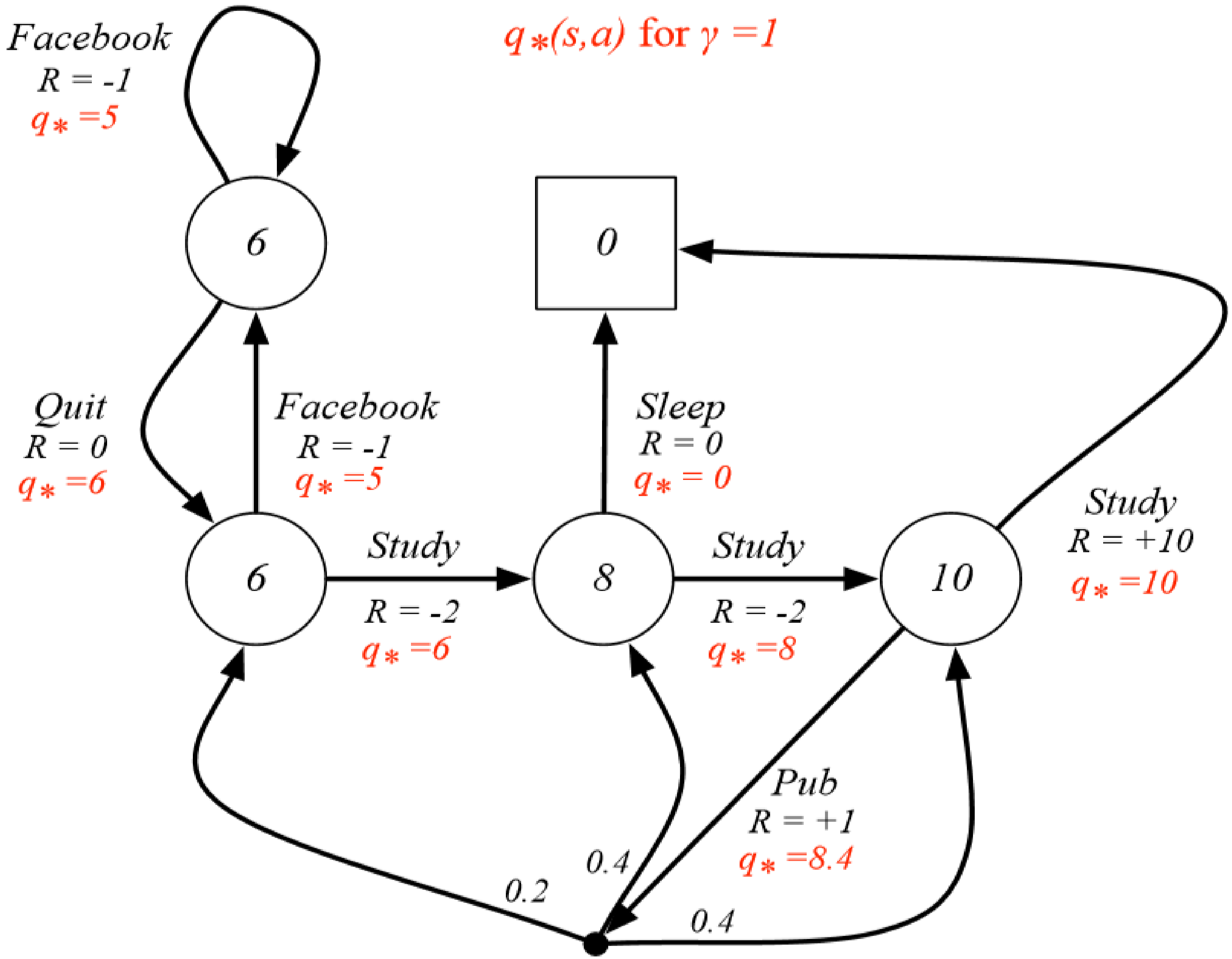
David Silver, Reinforcement Learning, PPT

Example: Optimal Value Function for Student MDP



David Silver, Reinforcement Learning, PPT

Example: Optimal Action-Value Function for Student MDP



David Silver, Reinforcement Learning, PPT

Finding an Optimal Policy

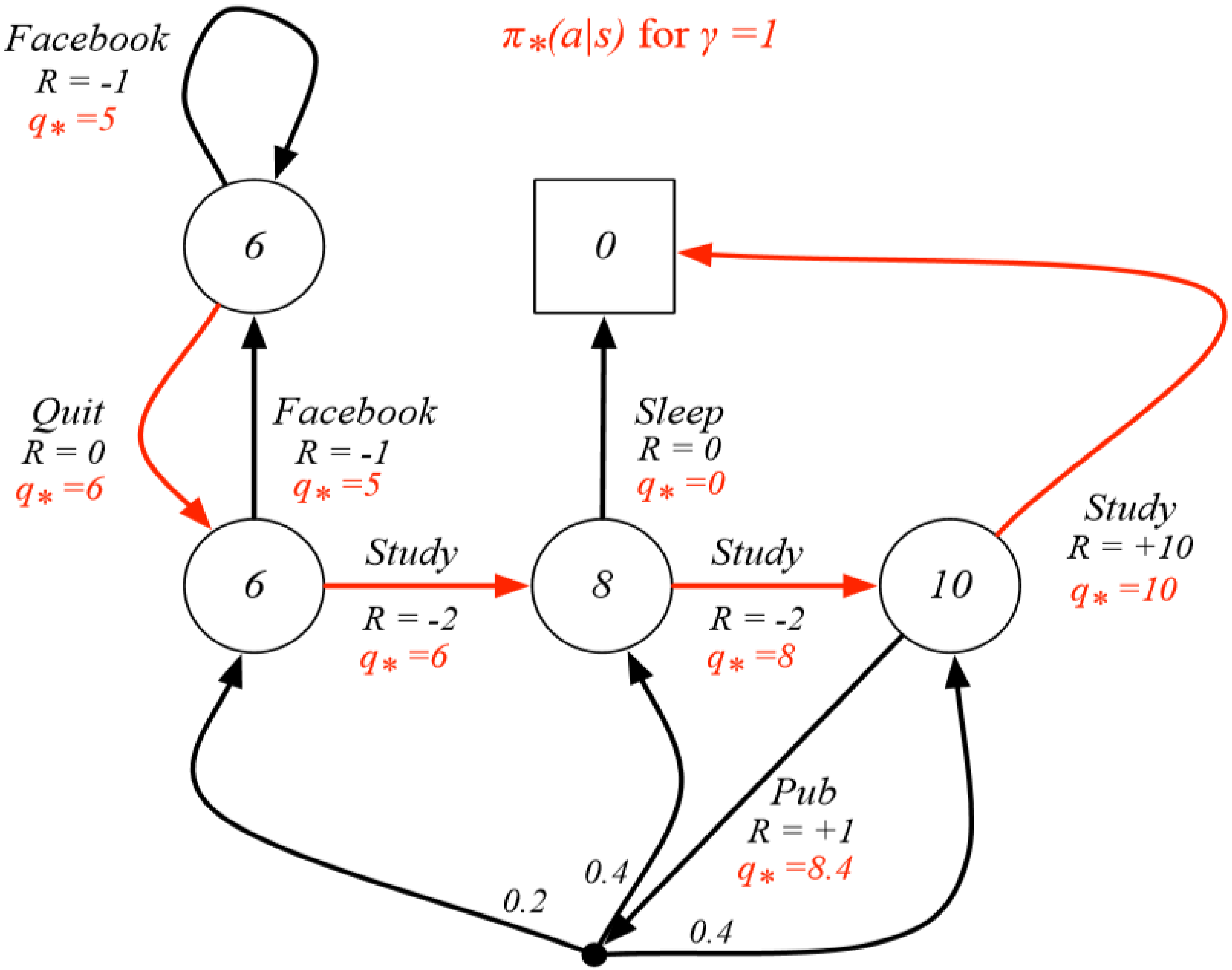
An optimal policy can be found by maximising over $q_*(s, a)$,

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

- There is always a deterministic optimal policy for any MDP
- If we know $q_*(s, a)$, we immediately have the optimal policy

David Silver, Reinforcement Learning, PPT

Example: Optimal Policy for Student MDP



David Silver, Reinforcement Learning, PPT

벨만 최적 방정식

- 일반 정책일 때 가치함수와 큐함수 사이의 관계

$$v_\pi(s) = \boldsymbol{E}_{a\sim\pi}[q_\pi(s,a)|S_t = s]$$

- 최적의 정책일 때 가치함수와 큐함수 사이의 관계

$$v^*(s) = \max_a[q^*(s,a)|S_t = s]$$

$$v^*(s) = \max_a \boldsymbol{E}[R_{t+1} + \gamma v^*(S_{t+1})|S_t = s]$$

벨만 최적 방정식

- 벨만 최적 방정식 : 행동을 선택할 때는 max, 가치함수 or 큐함수도 최적

- 가치함수에 대한 벨만 최적 방정식

$$v^*(s) = \max_a \boldsymbol{E}[R_{t+1} + \gamma v^*(S_{t+1}) | S_t = s, A_t = a]$$

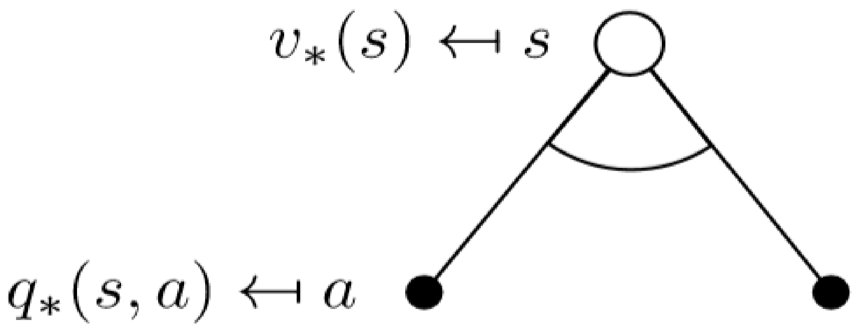
- 큐함수에 대한 벨만 최적 방정식

$$q^*(s,a) = \boldsymbol{E}[R_{t+1} + \gamma max_{a'} \; q^*(S_{t+1}, a') | S_t = s, A_t = a]$$

- 위의 두 식에서 R_{t+1} 앞에 기대값 $\boldsymbol{E}$가 있는 이유는 에이전트가 행동을 한 후에 받는 보상은 에이전트가 선택하는 것이 아니고 환경이 주는 값이기 때문임

Bellman Optimality Equation for v∗

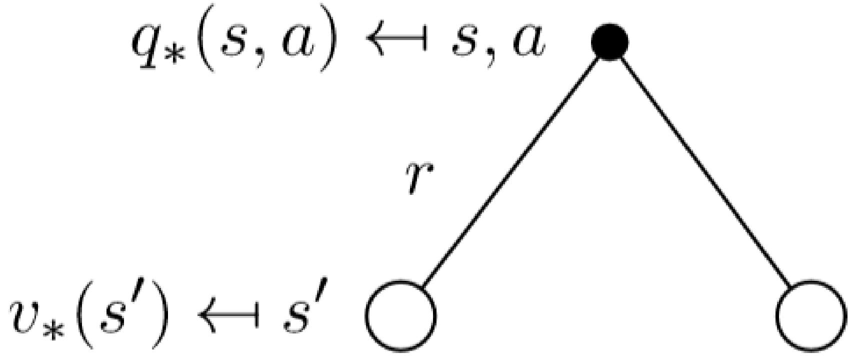
The optimal value functions are recursively related by the Bellman optimality equations:



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

David Silver, Reinforcement Learning, PPT

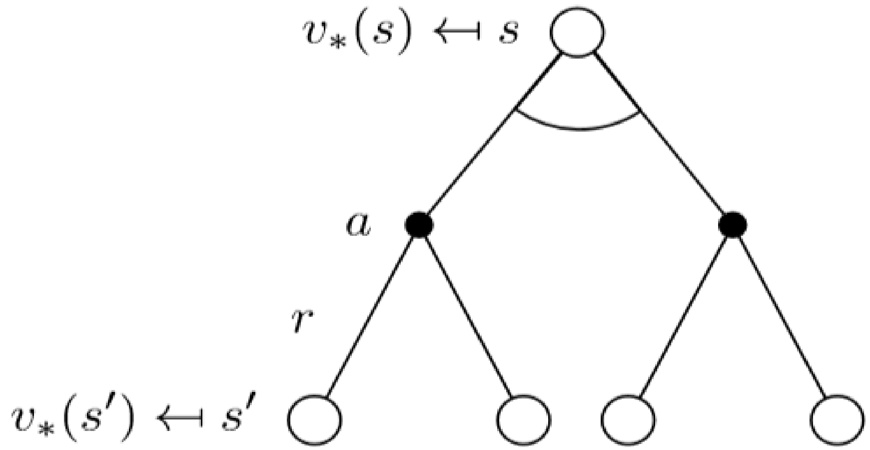
Bellman Optimality Equation for Q*



$$q_*(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

David Silver, Reinforcement Learning, PPT

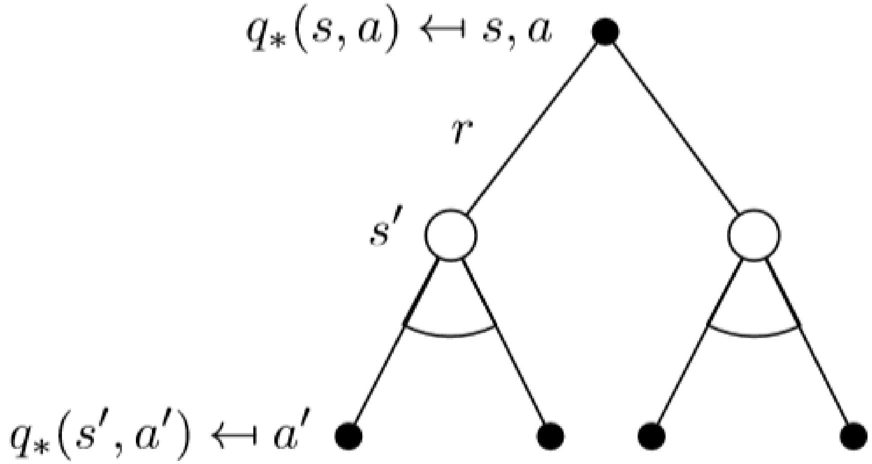
Bellman Optimality Equation for V∗ (2)



$$v_*(s) = \max_a \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

David Silver, Reinforcement Learning, PPT

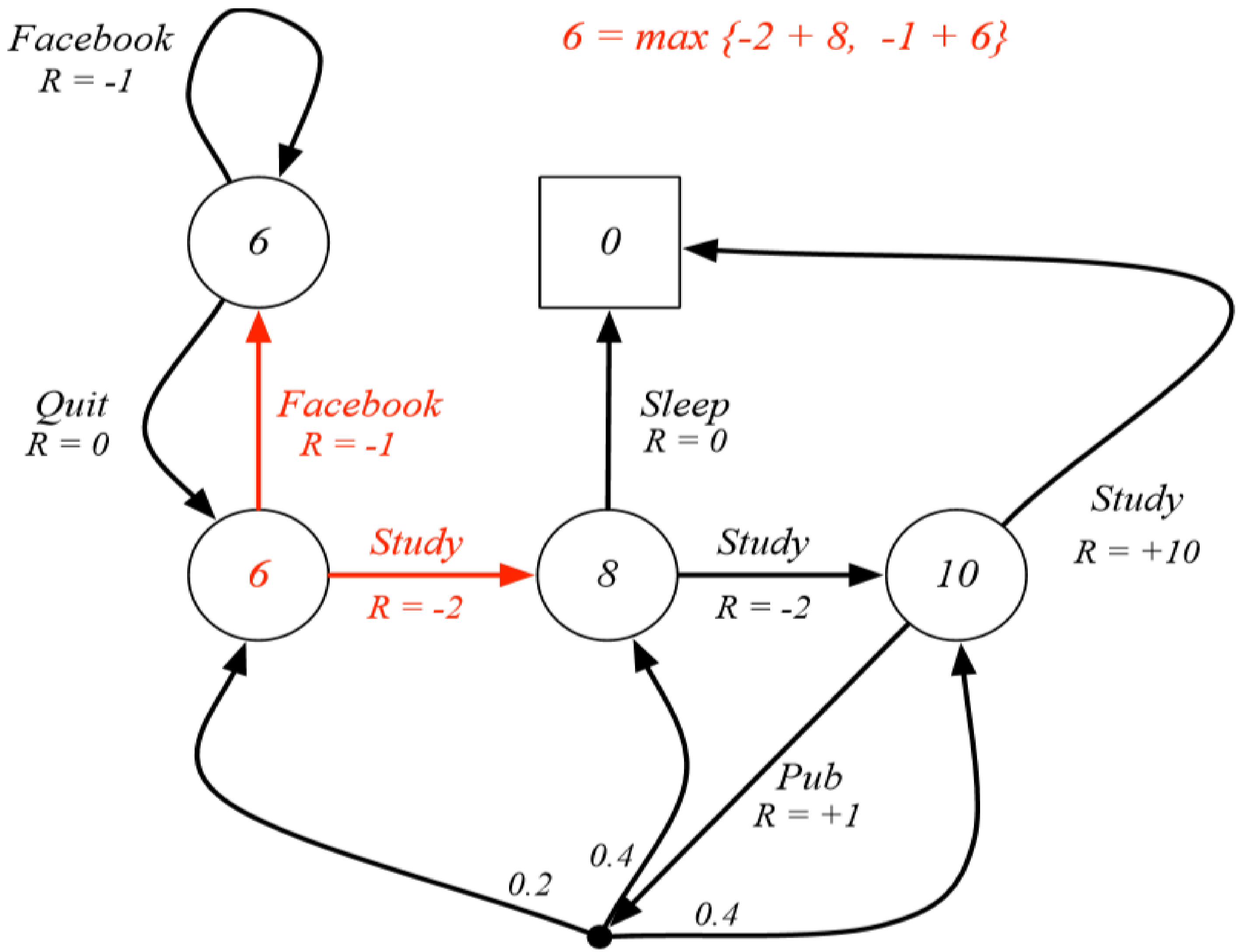
Bellman Optimality Equation for Q∗ (2)



$$q_*(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \max_{a'} q_*(s',a')$$

David Silver, Reinforcement Learning, PPT

Example: Bellman Optimality Equation in Student MDP



David Silver, Reinforcement Learning, PPT

Solving the Bellman Optimality Equation

- Bellman Optimality Equation is non-linear
- No closed form solution (in general)
- Many iterative solution methods
  - Value Iteration
  - Policy Iteration
  - Q-learning
  - Sarsa

David Silver, Reinforcement Learning, PPT

Summary

1. 보상의 표현

- 반환값(Return)

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_{t+T}$$

- 가치함수

$$v(s) = \boldsymbol{E}[G_t | S_t = s]$$

2. 벨만 기대 방정식 (Bellman Expectation Eqn.)

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

$$q_\pi(s,a) = \boldsymbol{E_\pi}[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t = s, A_t = a]$$

3. 벨만 최적 방정식 (Bellman Optimality Eqn.)

$$v^*(s) = \max_a \boldsymbol{E}[R_{t+1} + \gamma v^*(S_{t+1}) | S_t = s, A_t = a]$$

$$q^*(s,a) = \boldsymbol{E}[R_{t+1} + \gamma max_{a'}\ q^*(S_{t+1}, a') | S_t = s, A_t = a]$$