Lecture 2: Markov Decision Processes

David Silver

Introduction to MDPs

RL에서의 환경을 표현. env가 모두 다 관측 가능한 상황

- *Markov decision processes* formally describe an environment for reinforcement learning
- Where the environment is *fully observable*
- i.e. The current *state* completely characterises the process
- Almost all RL problems can be formalised as MDPs, e.g.
  - Optimal control primarily deals with continuous MDPs
  - Partially observable problems can be converted into MDPs
  - Bandits are MDPs with one state

Markov Property

"The future is independent of the past given the present"

Definition

A state S_t is *Markov* if and only if

S가 필요한 정보를 모두 가지고 있어 history가 필요하지 않다.

$$\mathbb{P}[S_{t+1} \mid S_t] = \mathbb{P}[S_{t+1} \mid S_1, ..., S_t]$$

- The state captures all relevant information from the history
- Once the state is known, the history may be thrown away
- i.e. The state is a sufficient statistic of the future

State Transition Matrix

MP, MRP, MDP 에 모두 사용됨. env 을 설명함

For a Markov state s and successor state s', the *state transition probability* is defined by

→ 액션이 없음

< s에서 s'로 갈 확률 >

$$\mathcal{P}_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s\right]$$

State transition matrix $\mathcal{P}$ defines transition probabilities from all states s to all successor states s',

$$\mathcal{P} = \text{from} \begin{bmatrix} & \text{to} & \\ \mathcal{P}_{11} & \dots & \mathcal{P}_{1n} \\ \vdots & & \\ \mathcal{P}_{n1} & \dots & \mathcal{P}_{nn} \end{bmatrix}$$

→ 합이 1. s = 1 에서 다른 s'로 갈 확률의 합은 1이어야 하지.

where each row of the matrix sums to 1.

Markov Process

S는 n 개 존재

다음 state로의 이동은 랜덤. P에 따라 랜덤.

A Markov process is a memoryless random process, i.e. a sequence of random states $S_1, S_2, \ldots$ with the Markov property.

과거기억X

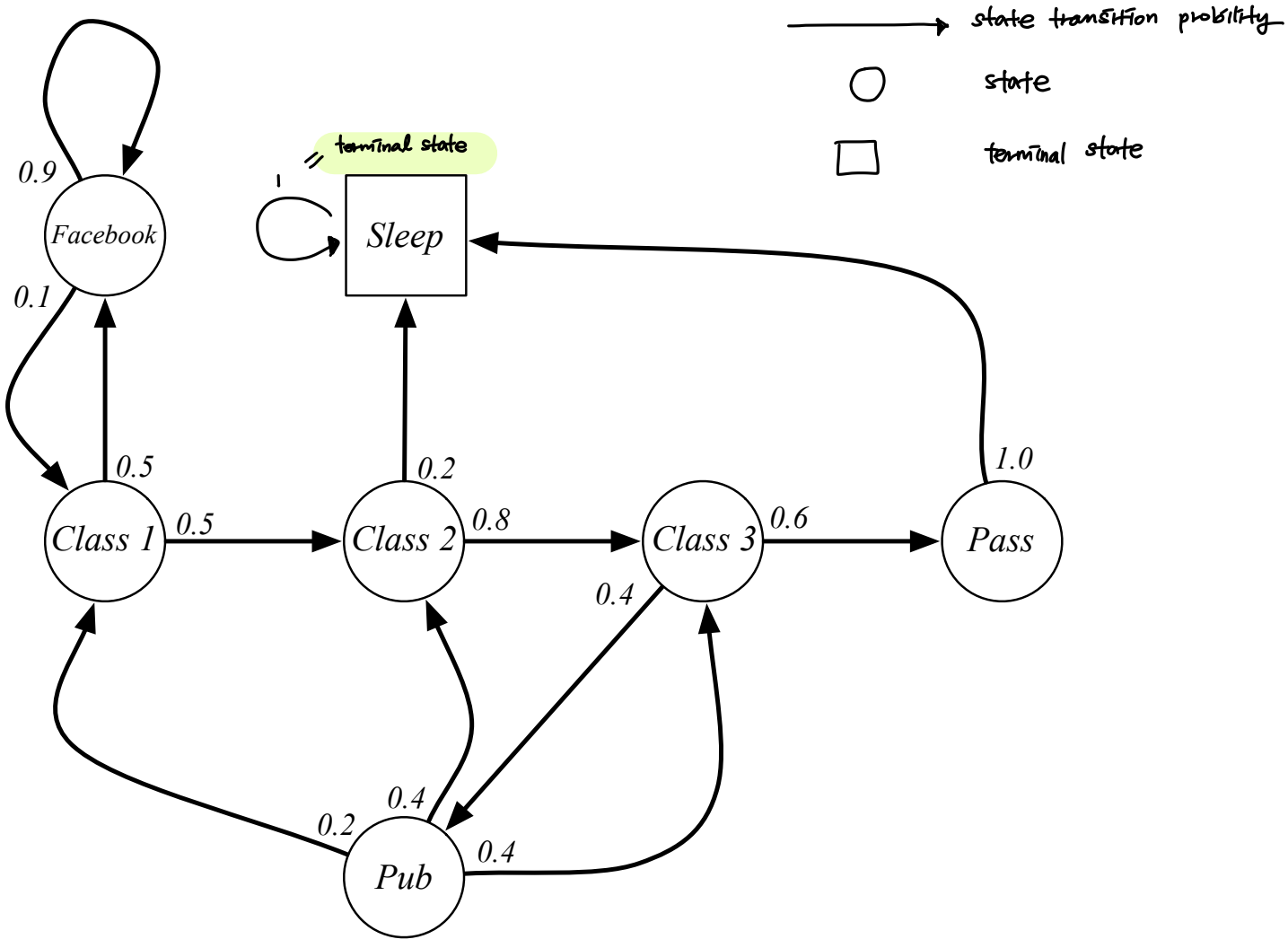
샘플링이 가능하다. 같은 state 에서 시작해도 다른 sequence 들 가질수있다

Definition

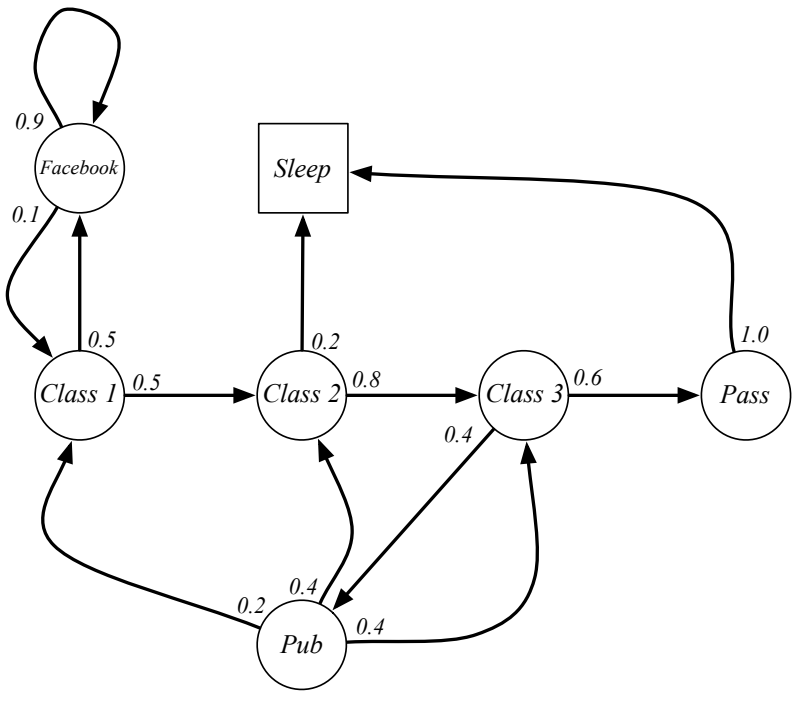
A *Markov Process* (or *Markov Chain*) is a tuple $\langle \mathcal{S}, \mathcal{P} \rangle$

- $\mathcal{S}$ is a (finite) set of states
- $\mathcal{P}$ is a state transition probability matrix,
  $\mathcal{P}_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s\right]$

Example: Student Markov Chain



Example: Student Markov Chain Episodes



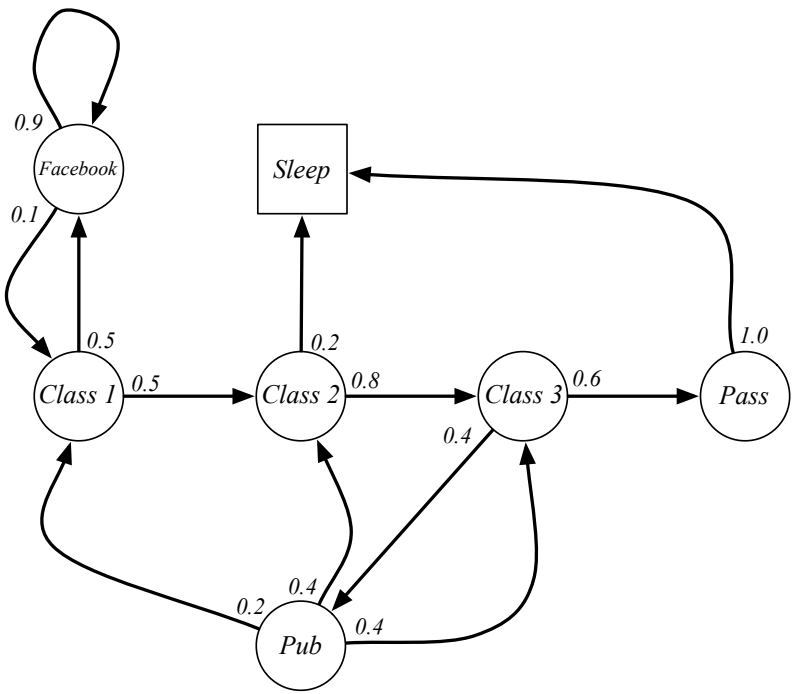
어떤한 S 에서 시작해서 종료 State 까지 가는 일련의 sequence

Sample episodes for Student Markov Chain starting from $S_1 = \text{C1}$

$$S_1, S_2, ..., S_T$$

- C1 C2 C3 Pass Sleep
- C1 FB FB C1 C2 Sleep
- C1 C2 C3 Pub C2 C3 Pass Sleep
- C1 FB FB C1 C2 C3 Pub C1 FB FB FB C1 C2 C3 Pub C2 Sleep

Example: Student Markov Chain Transition Matrix



$$\mathcal{P} = \begin{array}{c} \\ C1 \\ C2 \\ C3 \\ Pass \\ Pub \\ FB \\ Sleep \end{array} \begin{array}{c} \begin{array}{ccccccc} C1 & C2 & C3 & Pass & Pub & FB & Sleep \end{array} \\ \begin{bmatrix} & 0.5 & & & & 0.5 & \\ & & 0.8 & & & & 0.2 \\ & & & 0.6 & 0.4 & & \\ & & & & & & 1.0 \\ 0.2 & 0.4 & 0.4 & & & & \\ 0.1 & & & & & 0.9 & \\ & & & & & & 1 \end{bmatrix} \end{array}$$

P와 S가 모두 있으므로

MP가 완전히 정의되었다.

Markov Reward Process

A Markov reward process is a Markov chain with values.

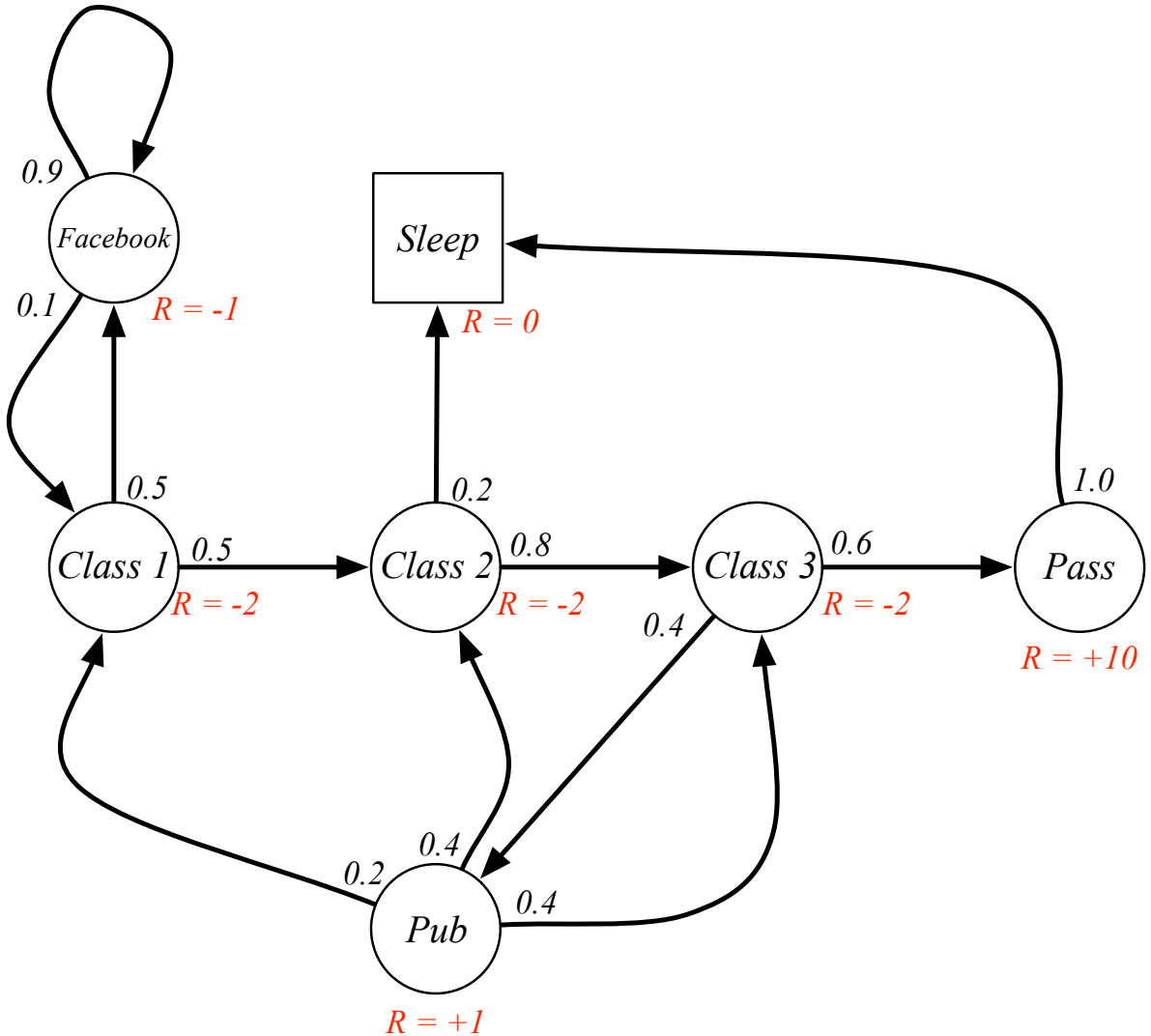
Definition

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, $\gamma \in [0, 1]$

* reward function : 어떤 state 에 도착하면 reward 값을 준다

Example: Student MRP



Return

RL은 Return을 maximize 하고자 한다

Definition

지금 state부터 미래의 보상까지 예측하여 더한 값.

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 순간적인 값을 중요시
  - γ close to 1 leads to "far-sighted" evaluation 장기적으로 중요시.

Why discount?

Most Markov reward and decision processes are discounted. Why?

- Mathematically convenient to discount rewards
  수학적인 이유는 계산하기 편해서
- 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.

  에피소드가 무조건 끝나도록 보장되면 discount 할 필요가 없다

Value Function

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

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

↳ state s 에 왔을 때 return 의 기댓값

Example: Student MRP Returns

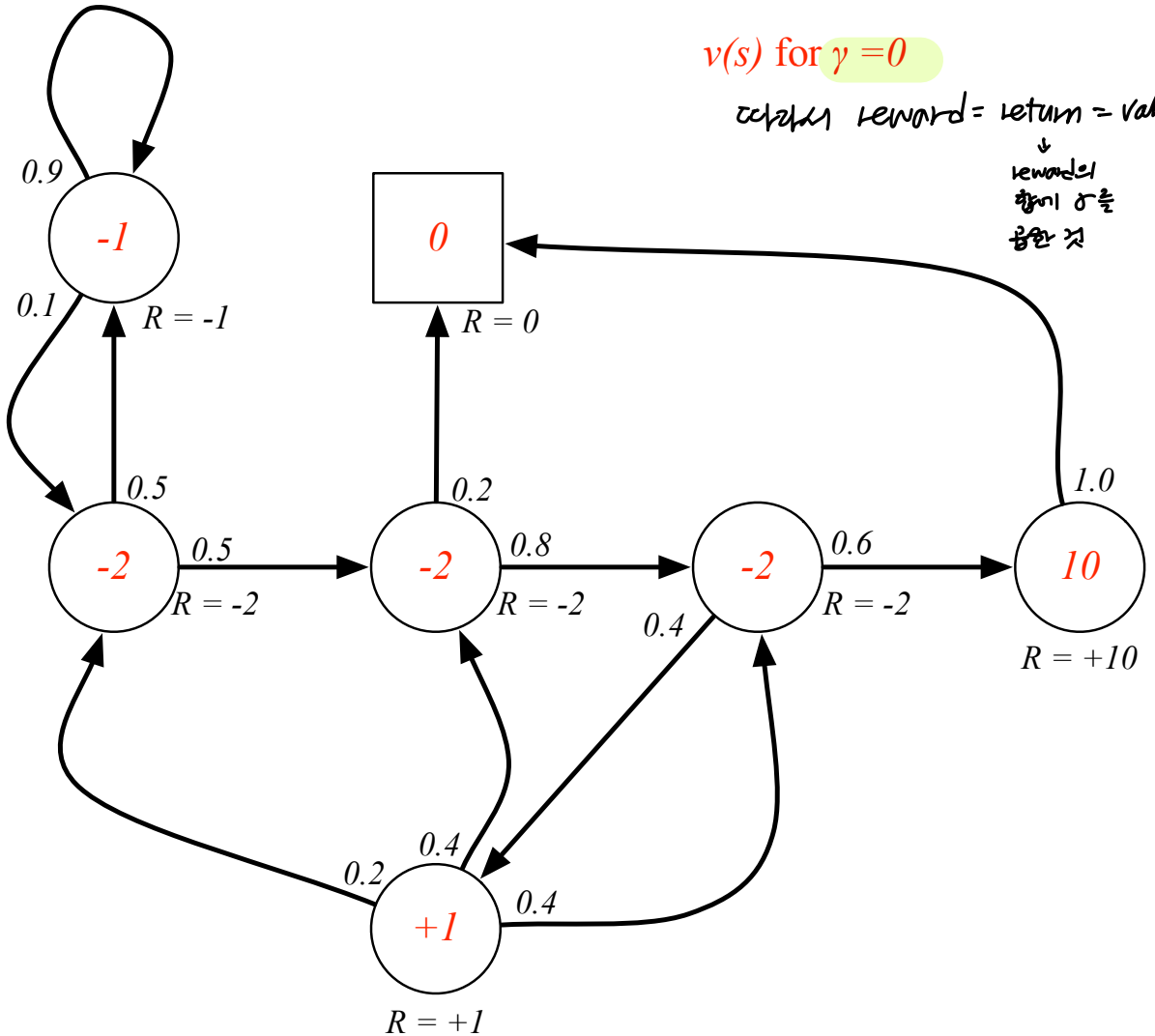
Sample returns for Student MRP:
Starting from $S_1 = \text{C1}$ with $\gamma = \frac{1}{2}$

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

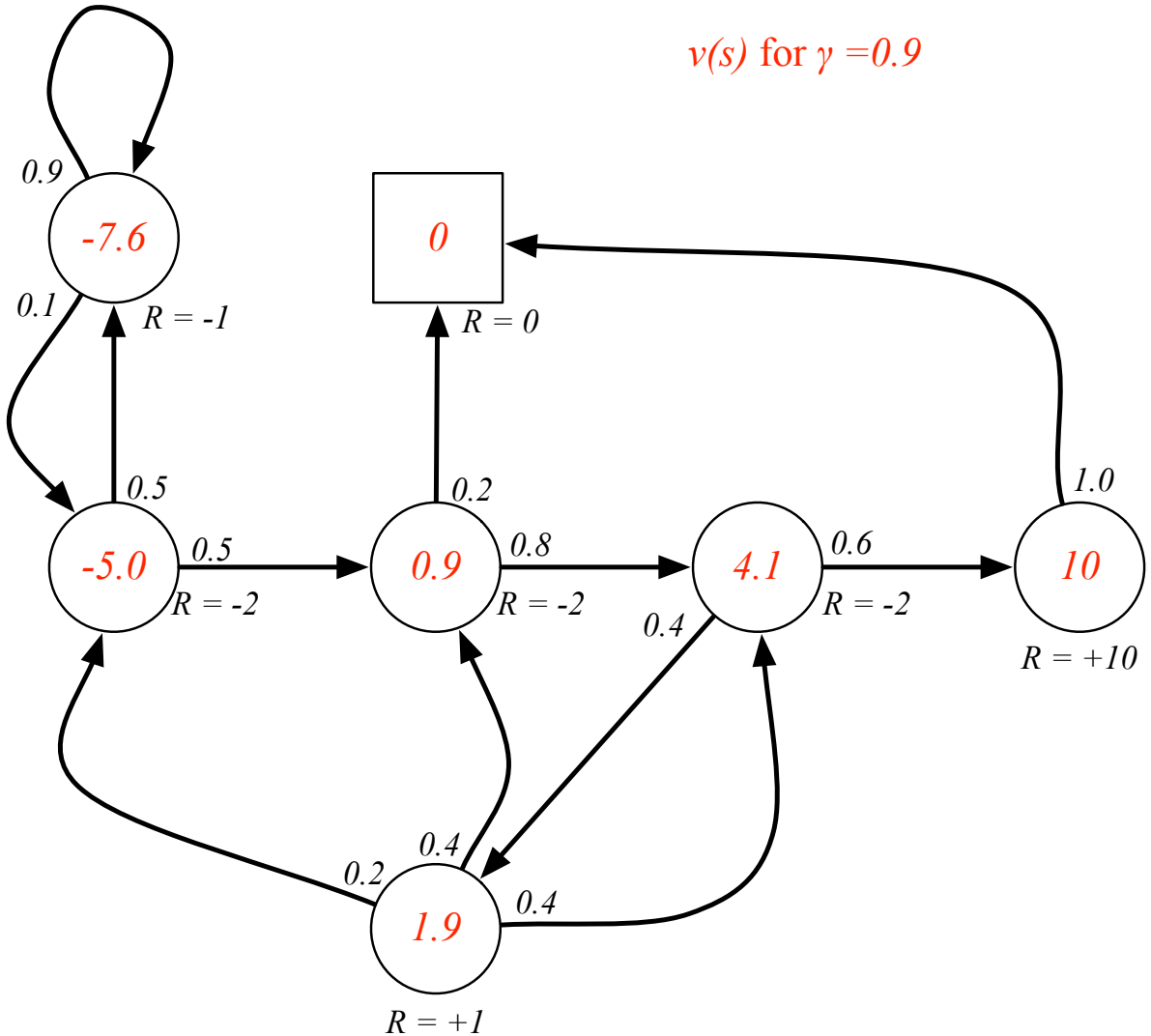
C1 C2 C3 Pass Sleep	$v_1 = -2 - 2 * \frac{1}{2} - 2 * \frac{1}{4} + 10 * \frac{1}{8}$	=	-2.25
C1 FB FB C1 C2 Sleep	$v_1 = -2 - 1 * \frac{1}{2} - 1 * \frac{1}{4} - 2 * \frac{1}{8} - 2 * \frac{1}{16}$	=	-3.125
C1 C2 C3 Pub C2 C3 Pass Sleep	$v_1 = -2 - 2 * \frac{1}{2} - 2 * \frac{1}{4} + 1 * \frac{1}{8}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

확률에 따라 여러 샘플링 (에피소드)가 발생 → 이 에피소드들의 G 를 구하고 이를 평균냄.

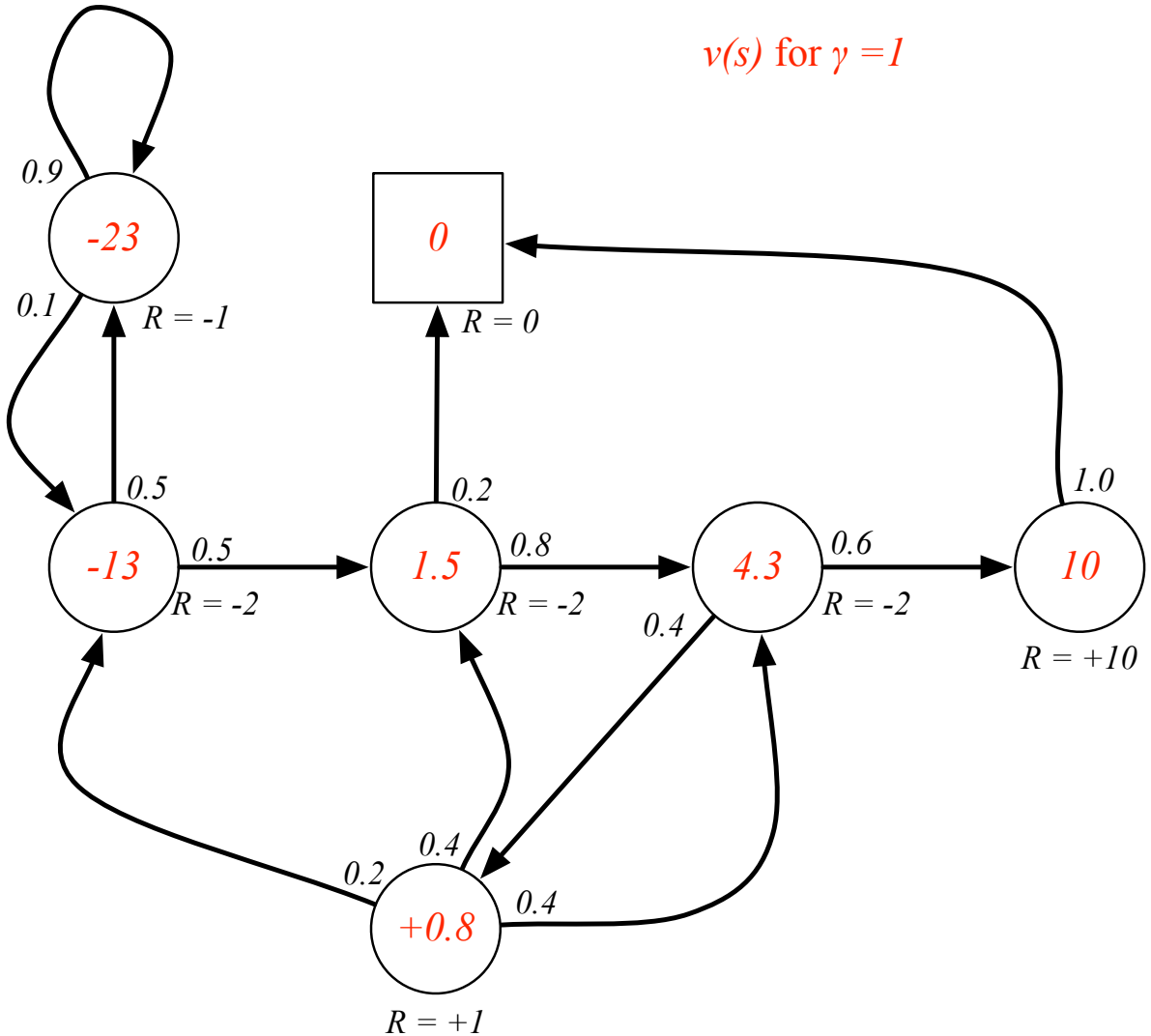
Example: State-Value Function for Student MRP (1)



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



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



Bellman Equation for MRPs

value function의 핵심!

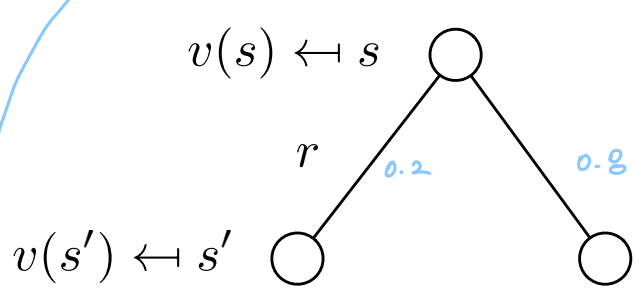
The value function can be decomposed into two parts:

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

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

Bellman Equation for MRPs (2)

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



$R_s = E[R_{t+1} \mid S_t = s]$

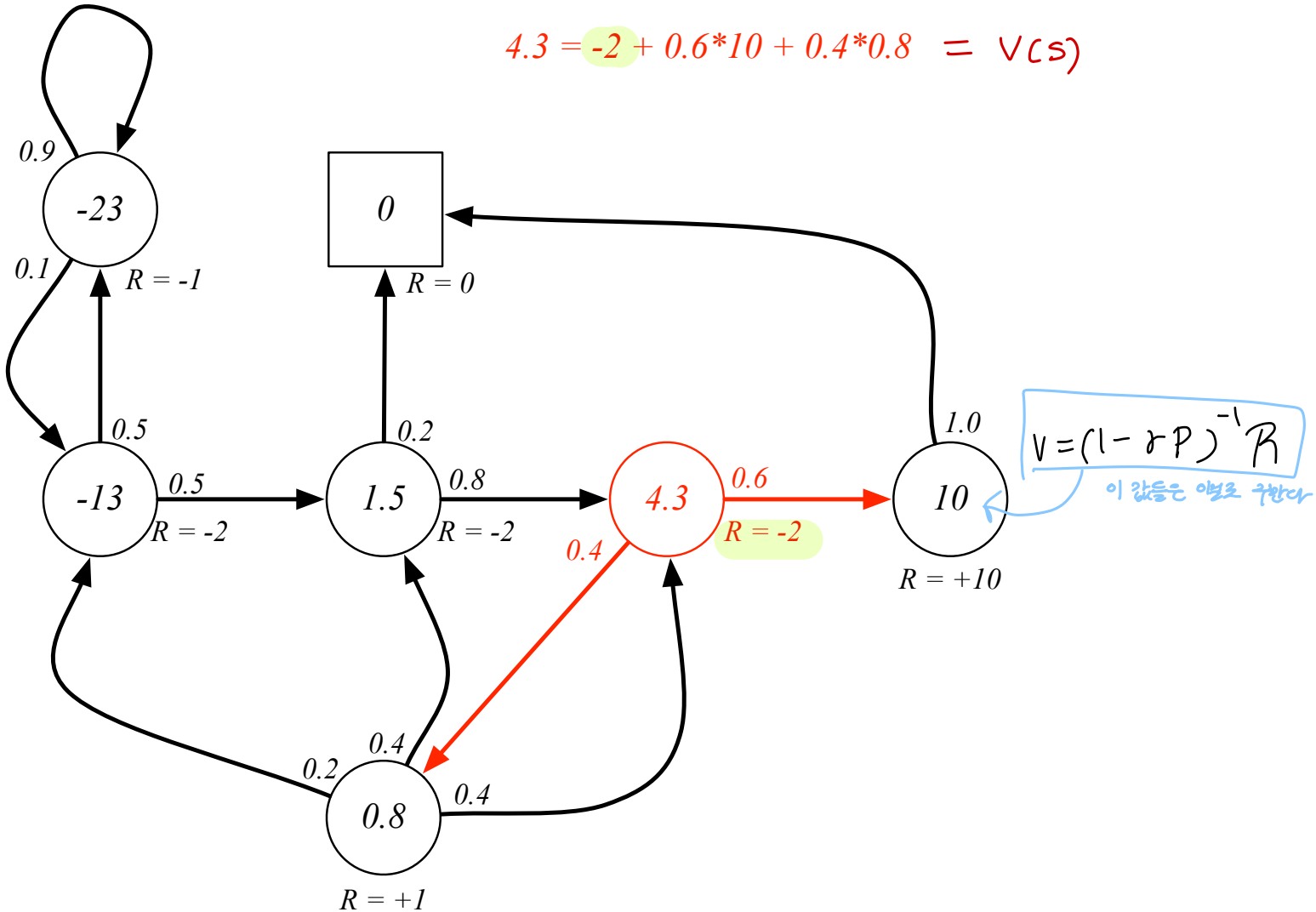
예제

← transition probability. ($\mathcal{P}_{ss'}$)

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

E (2가지 경우에 대한 평균)을 계산하면 이 식이 나온다.

Example: Bellman Equation for Student MRP



Bellman Equation in Matrix Form

The Bellman equation can be expressed concisely using matrices,

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

↓ vector와 matrice로 표현

$$v = \mathcal{R} + \gamma \mathcal{P} v$$

where v is a column vector with one entry per state

$$\begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix} = \begin{bmatrix} \mathcal{R}_1 \\ \vdots \\ \mathcal{R}_n \end{bmatrix} + \gamma \begin{bmatrix} \mathcal{P}_{11} & \dots & \mathcal{P}_{1n} \\ \vdots & & \\ \mathcal{P}_{11} & \dots & \mathcal{P}_{nn} \end{bmatrix} \begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix}$$

Solving the Bellman Equation

- The Bellman equation is a linear equation
- It can be solved directly:

$$
\begin{aligned}
v &= \mathcal{R} + \gamma \mathcal{P} v \\
(I - \gamma \mathcal{P})\, v &= \mathcal{R} \\
v &= (I - \gamma \mathcal{P})^{-1} \mathcal{R}
\end{aligned}
$$

- Computational complexity is $O(n^3)$ for n states
- Direct solution only possible for small MRPs 작은 문제에서 사용
- There are many iterative methods for large MRPs, e.g. 큰 문제에서는 아래 방법 사용
  - Dynamic programming
  - Monte-Carlo evaluation
  - Temporal-Difference learning

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.

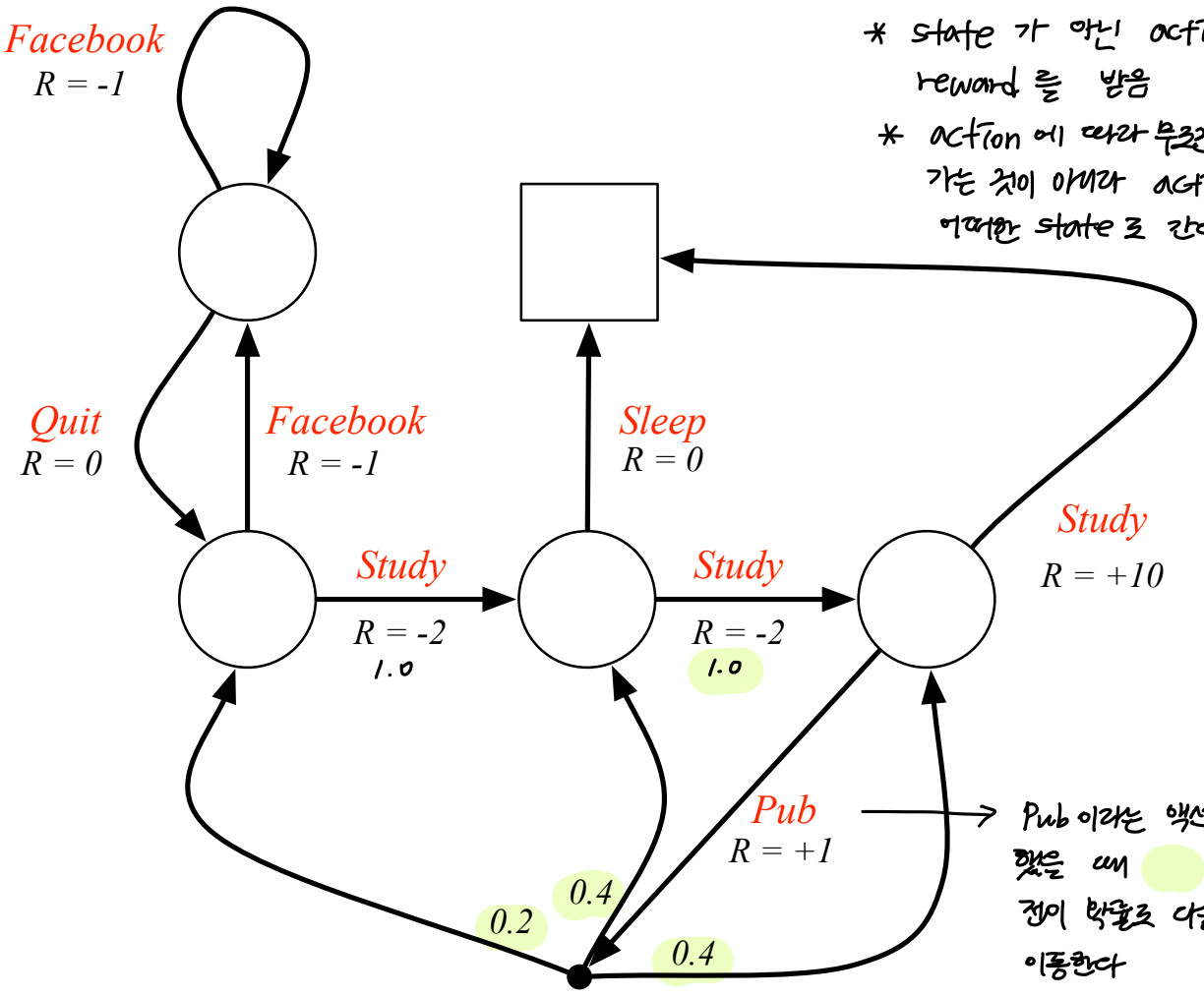
Definition

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, $\mathcal{P}^{a}_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a\right]$
- $\mathcal{R}$ is a reward function, $\mathcal{R}^{a}_{s} = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$
- γ is a discount factor $\gamma \in [0, 1]$.

Example: Student MDP

MDP 의 env



* State 의 이름이 없어짐.
* State 가 아닌 action에 따른 reward 를 받음
* action 에 따라 무조건 한 state로 가는 것이 아니라 action에 따른 확률로 어떠한 state 로 간다 (전이)

Pub 이라는 액션을 했을 때 0.2, 0.4, 0.4 와 같은 전이 확률로 다음 state로 이동한다

Policies (1)

action을 하기 위한 정책. agent의 행동을 완전히 결정

Definition

State S_t에 있을 때 action a를 할 확률

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

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

- 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),
  $A_t \sim \pi(\cdot|S_t), \forall t > 0$

Policies (2)

- Given an MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ and a policy π
- The state sequence $S_1, S_2, \ldots$ is a Markov process $\langle \mathcal{S}, \mathcal{P}^\pi \rangle$
- The state and reward sequence $S_1, R_2, S_2, \ldots$ is a Markov reward process $\langle \mathcal{S}, \mathcal{P}^\pi, \mathcal{R}^\pi, \gamma \rangle$
- where

이렇게 해서, MP로 나타낼수있다

* agent의 policy가 고정이 됐을 때

$$\mathcal{P}^\pi_{s,s'} = \sum_{a \in \mathcal{A}} \pi(a|s) \mathcal{P}^a_{ss'}$$

$$\mathcal{R}^\pi_s = \sum_{a \in \mathcal{A}} \pi(a|s) \mathcal{R}^a_s$$

(P)

MP는 원래 action 없이 transition probability만 존재
MDP에서 policy가 고정된다면, P^π를 구할수 있음.
s에서 어떠한 policy π에 따라 a를 할 확률과
그 a를 했을 때 s'을 갈 확률의 합이 곧
s에서 s'로 갈 transition probability 이다.

why? 한 state에서 여러 action이 가능하고,
action이 하나의 s'로만 가는 것이 아니기 때문

in MP $s \xrightarrow{P_{ss'}} s'$

in MDP $s \xrightarrow{\pi(a|s)} a \xrightarrow{P^a_{ss'}} s'$

$\mathcal{P}^\pi_{ss'}$

s에서 a를 할 확률 x 어떤 a를 했을 때 s'로 갈 확률

s에서 a를 할 확률 x 어떤 a를 했을 때 받을 R

Value Function

Definition

어떤 s에서 π를 따라 한 에피소드를 완료했을 때, 여러 에피소드의 G의 평균

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

s에 따라 다른 v 값
state 1개에 대한 값

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

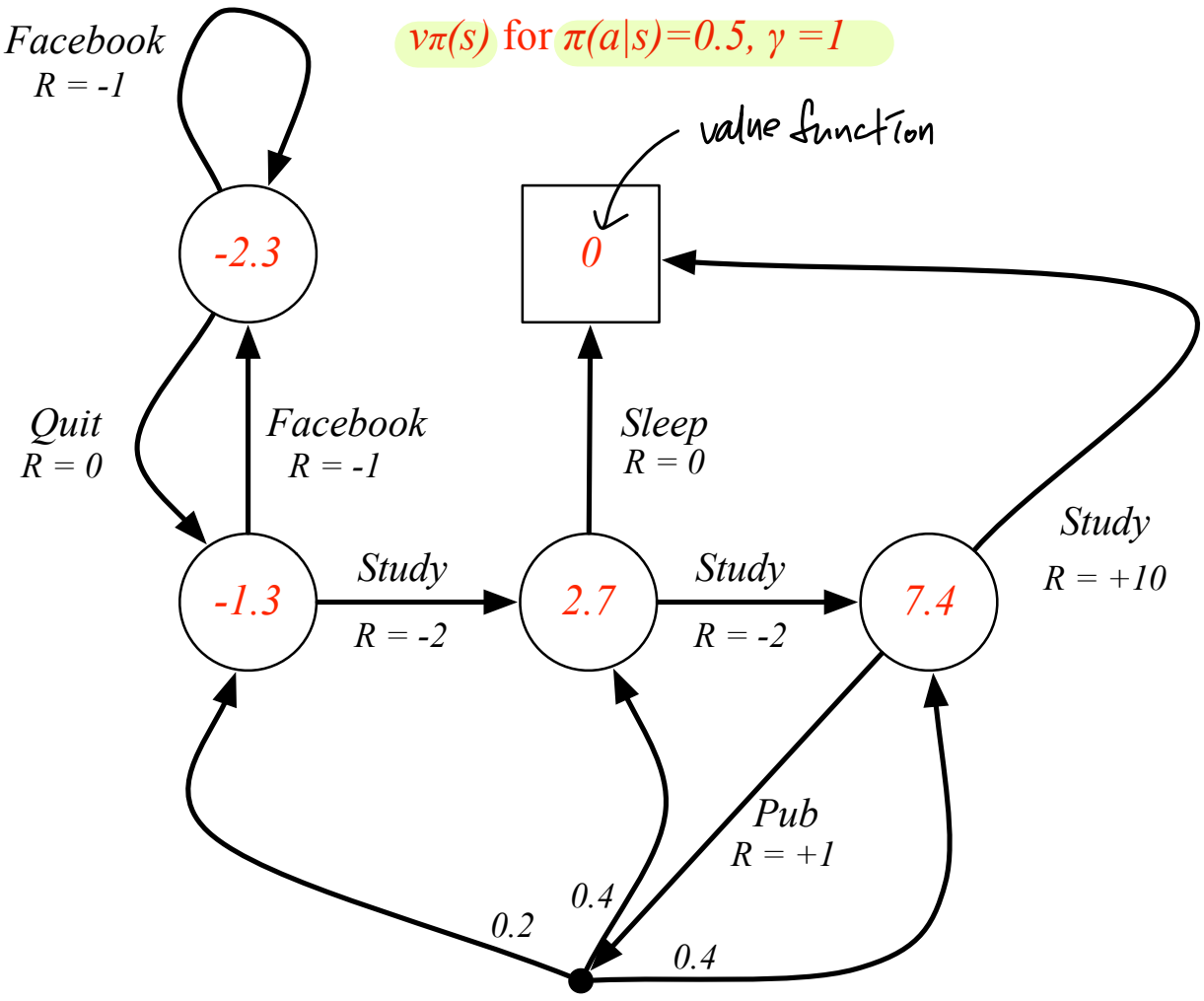
Definition

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

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

s & a 에 따라 다른 q 값.
액션 1개에 대한 값.

Example: State-Value Function for Student MDP



Bellman Expectation Equation

기댓값에 대해.

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,

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

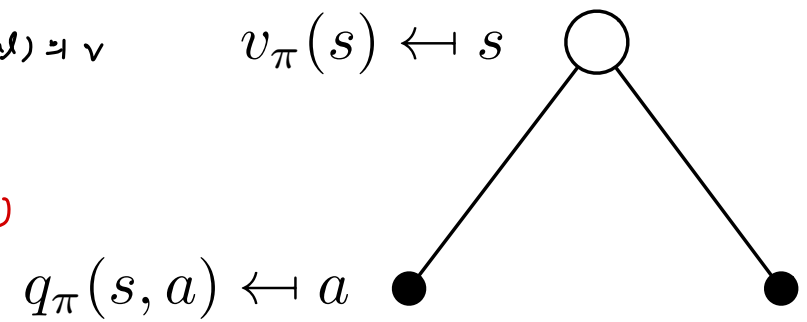
Bellman Expectation Equation for V^π

v_π(s) : s → 어떤 a → s' (terminal) 의 v

q_π(s,a) : s → a, → s' (terminal) 의 v

⋮

즉, $q_\pi(s,a)$ 에 액션을 할 확률 $\pi(a|s)$ 을 곱하여 더하면 $v_\pi(s)$ 과 같다.

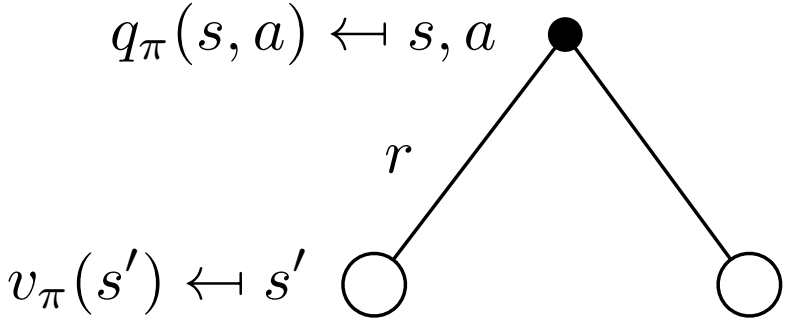


$v_\pi(s)$ 와 $q_\pi(s,a)$ 의 관계

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

s에서 a를 할 확률 X a를 했을 때 받을 v

Bellman Expectation Equation for Q^π



$q_\pi(s,a)$와 $v_\pi(s)$의 관계

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

s'에서 a를 했을 때 s'으로 갈 확률 × s'에서의 v

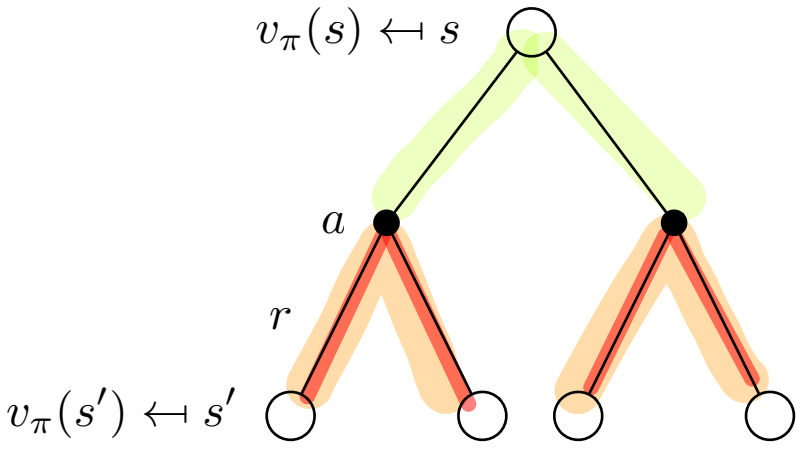
* 어떤 s에서 a를 한 상황.
BUT action 후 어떤 s'에 도착할지 모름. 따라서 존재하는 모든 s'에 대한 확률 $\mathcal{P}_{ss'}^a$와 그 곳에서의 $v_\pi(s')$를 곱해준다.

s에서 a를 했을 때 π를 따라서 게임을 끝까지 했을 an 얻을 return의 기댓값

s에서 a를 했을 때 얻을 $\mathcal{R}$의 평균값 $= \mathcal{R}_s^a$

a에 따라 어떤 state로 갈지 모르므로, 그 확률인 $\mathcal{P}_{ss'}^a$와 그 때 받을 미래 v인 $v_\pi(s')$의 곱.

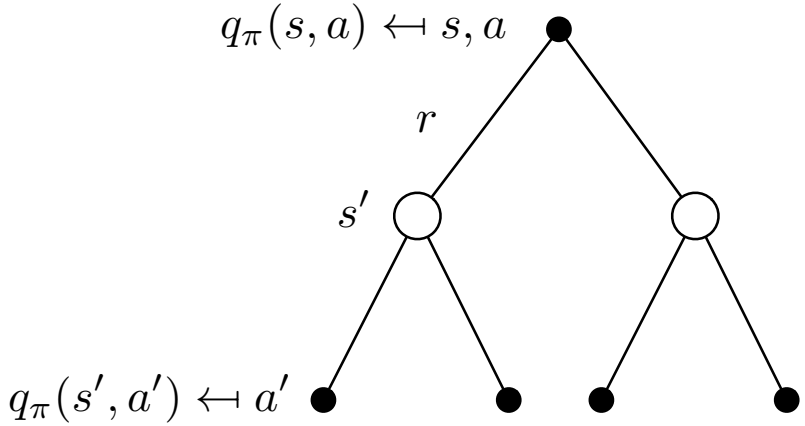
Bellman Expectation Equation for v_π (2)



$V_\pi(s)$를 $V_\pi(s)$로만 정리 (나중에 식 정리해줘고)

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

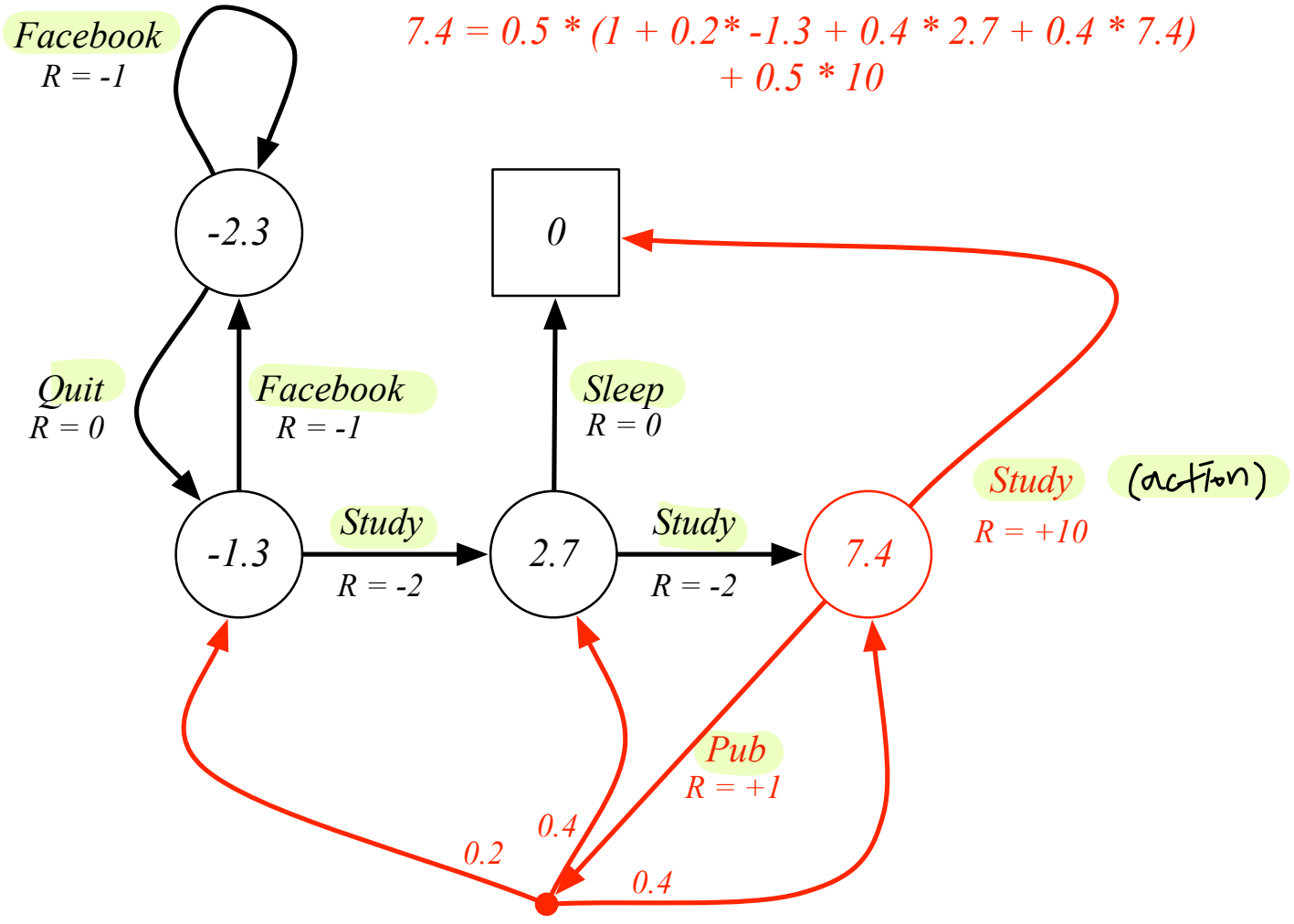
Bellman Expectation Equation for q_π (2)



q를 q로만 정의

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

Example: Bellman Expectation Equation in Student MDP



Bellman Expectation Equation (Matrix Form)

BUT 커지면 못푼다.

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

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

Optimal Value Function

Definition

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

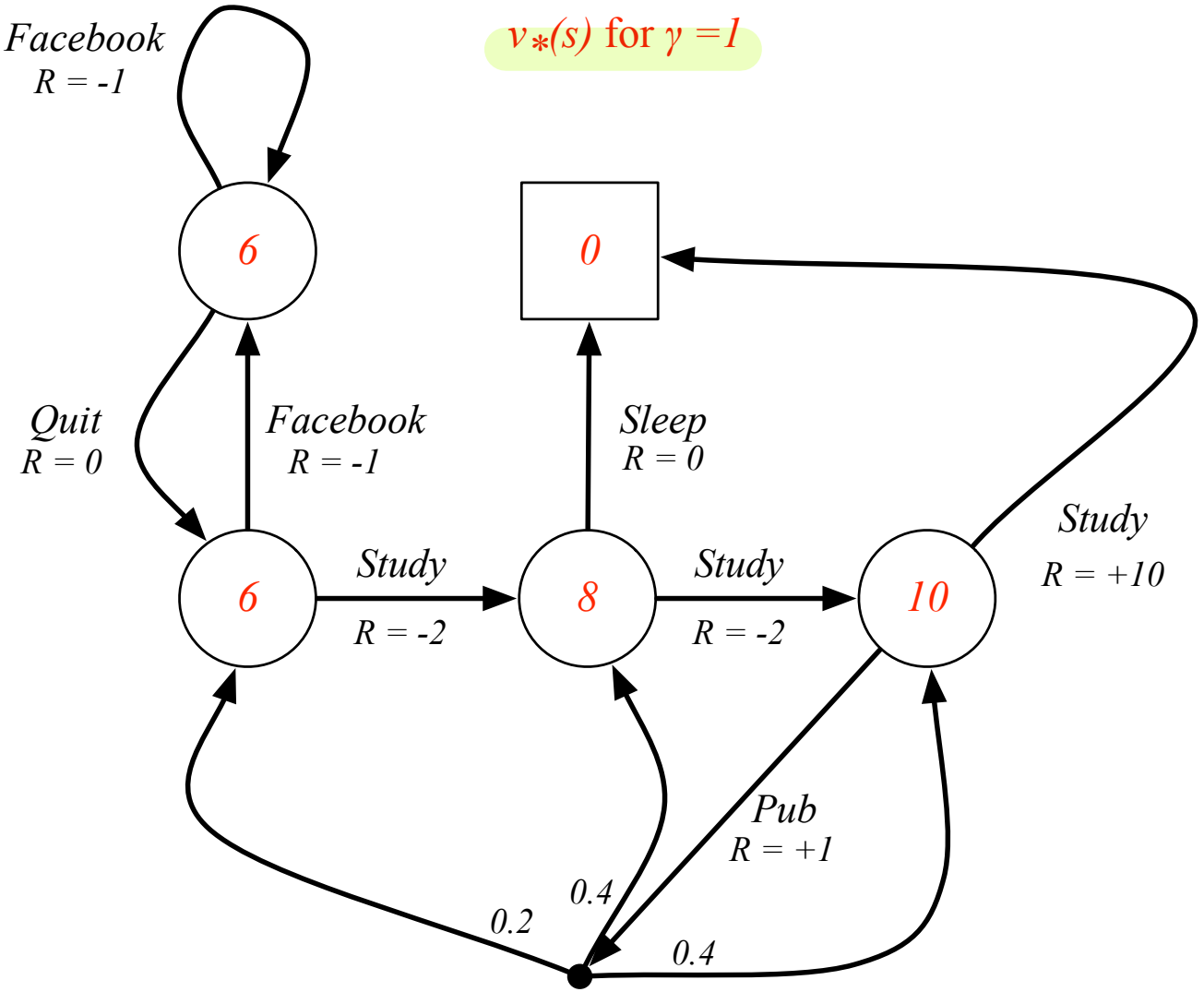
$$v_*(s) = \max_\pi v_\pi(s)$$

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

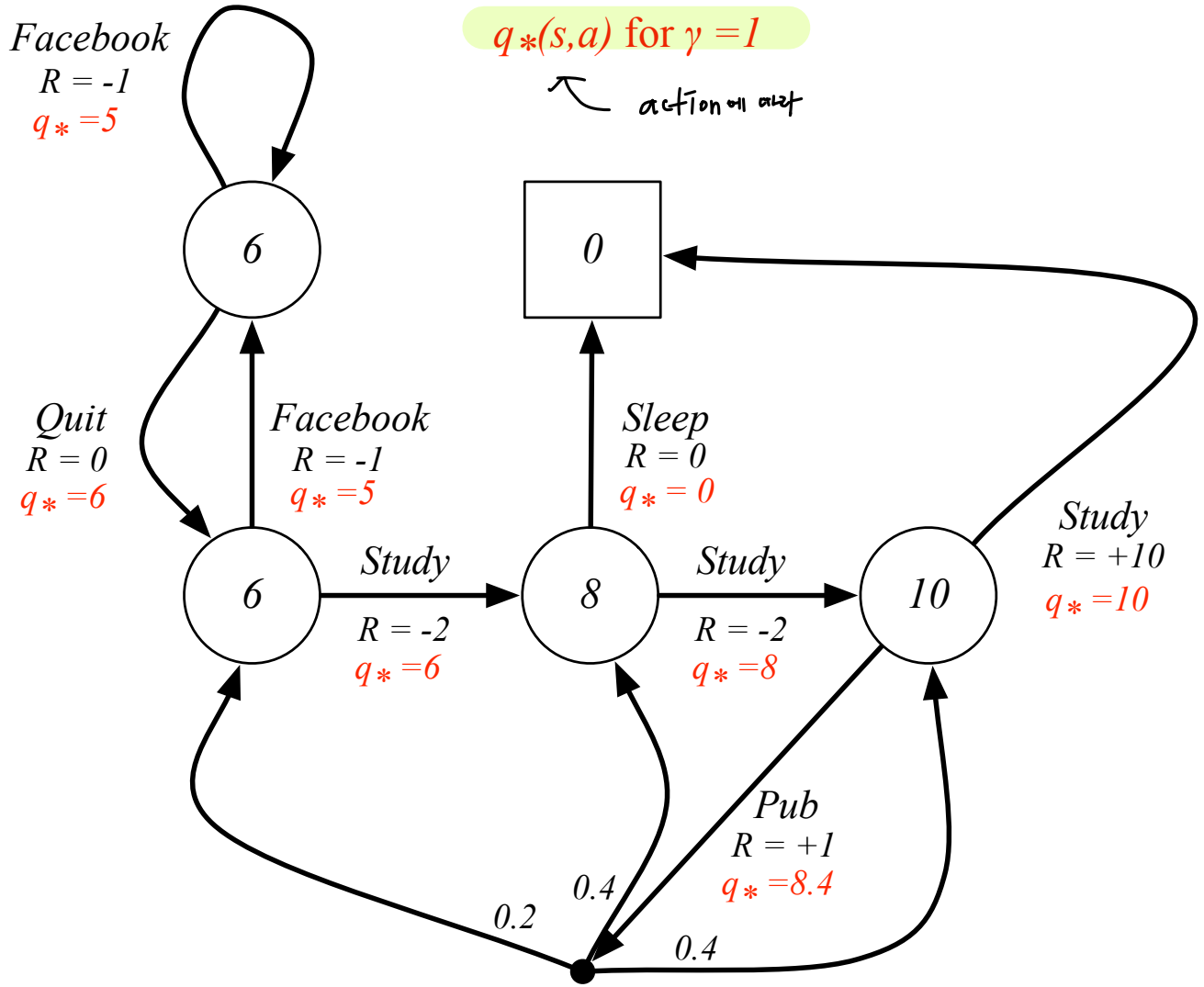
$$q_*(s, a) = \max_\pi q_\pi(s, a)$$

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

Example: Optimal Value Function for Student MDP



Example: Optimal Action-Value Function for Student MDP



Optimal Policy

Define a partial ordering over policies

어떤 두 개의 policy가 주어졌을 때, 항상 비교가 가능한 것은 아니다
단지, 비교가 가능할 때가 있다 → 밑에 있는 식

$$\pi \geq \pi' \text{ if } v_\pi(s) \geq v_{\pi'}(s), \forall s$$

Theorem

For any Markov Decision Process

- *There exists an optimal policy* π_* *that is better than or equal to all other policies,* $\pi_* \geq \pi, \forall \pi$ optimal policy는 무조건 존재
- *All optimal policies achieve the optimal value function,* $v_{\pi_*}(s) = v_*(s)$ π를 따라간 $V_\pi(S)$와 optimal value function $V(S)$는 같다
- *All optimal policies achieve the optimal action-value function,* $q_{\pi_*}(s, a) = q_*(s, a)$

Finding an Optimal Policy

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

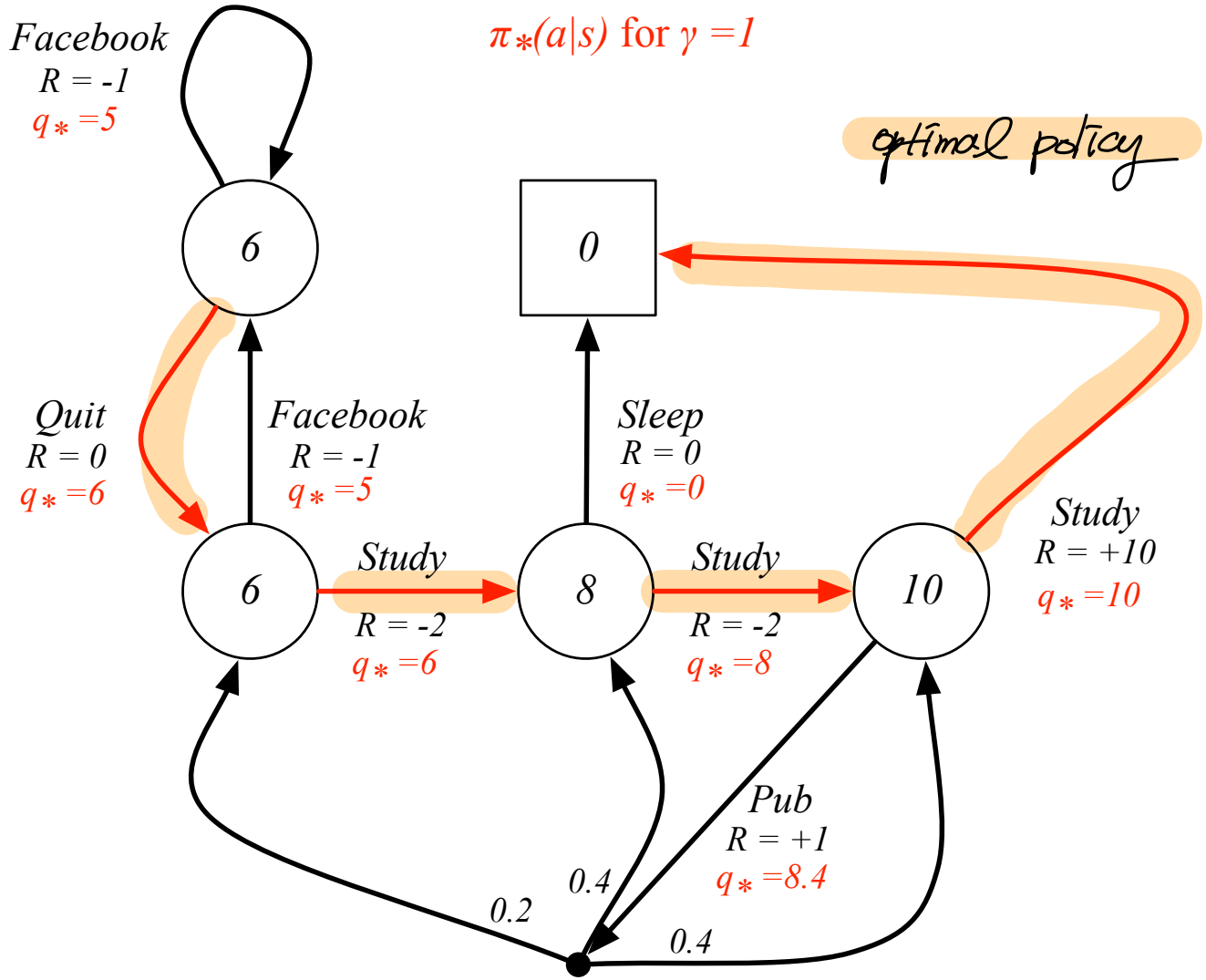
$q_*(s,a)$를 아는 순간, optimal policy가 무조건 존재

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

무조건 이쪽으로 가기 때문에

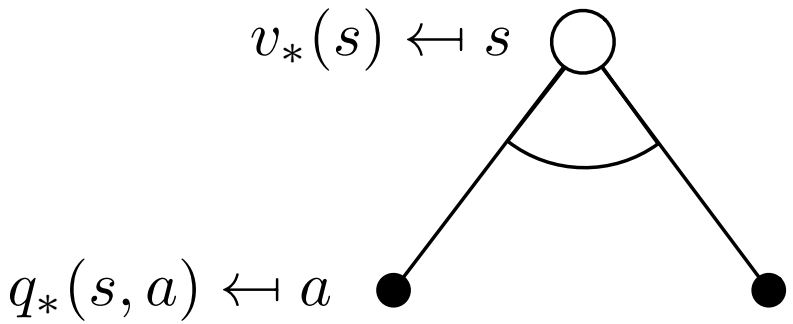
- There is always a deterministic optimal policy for any MDP
- If we know $q_*(s, a)$, we immediately have the optimal policy

Example: Optimal Policy for Student MDP



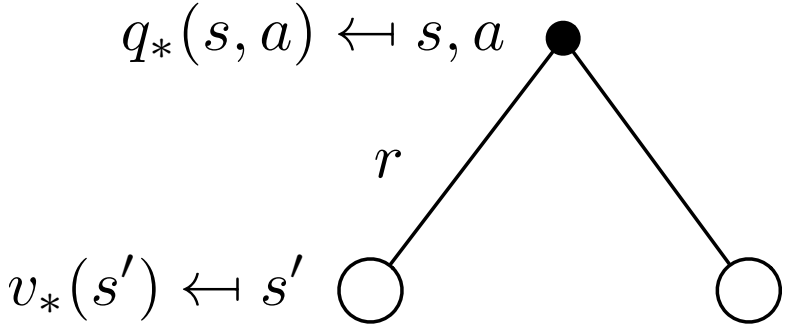
Bellman Optimality Equation for v_*

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



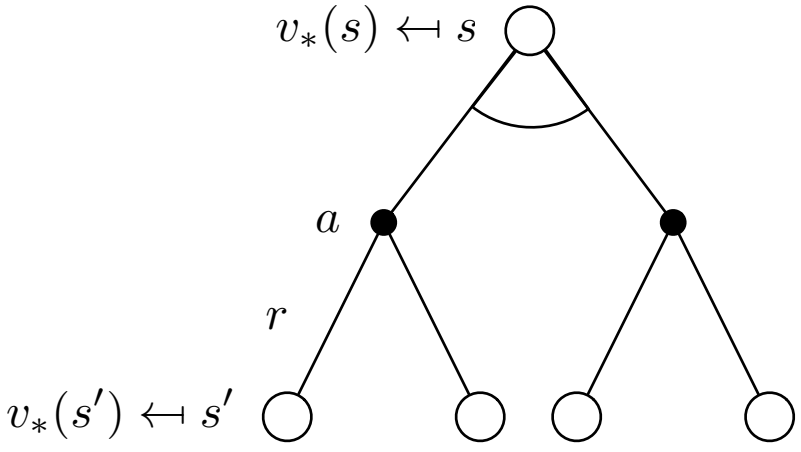
$$v_*(s) = \max_a q_*(s,a)$$

Bellman Optimality Equation for Q^*



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

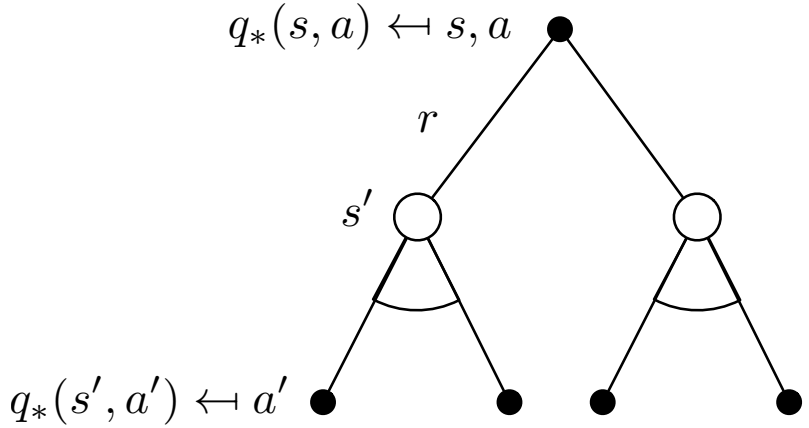
Bellman Optimality Equation for V^* (2)



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

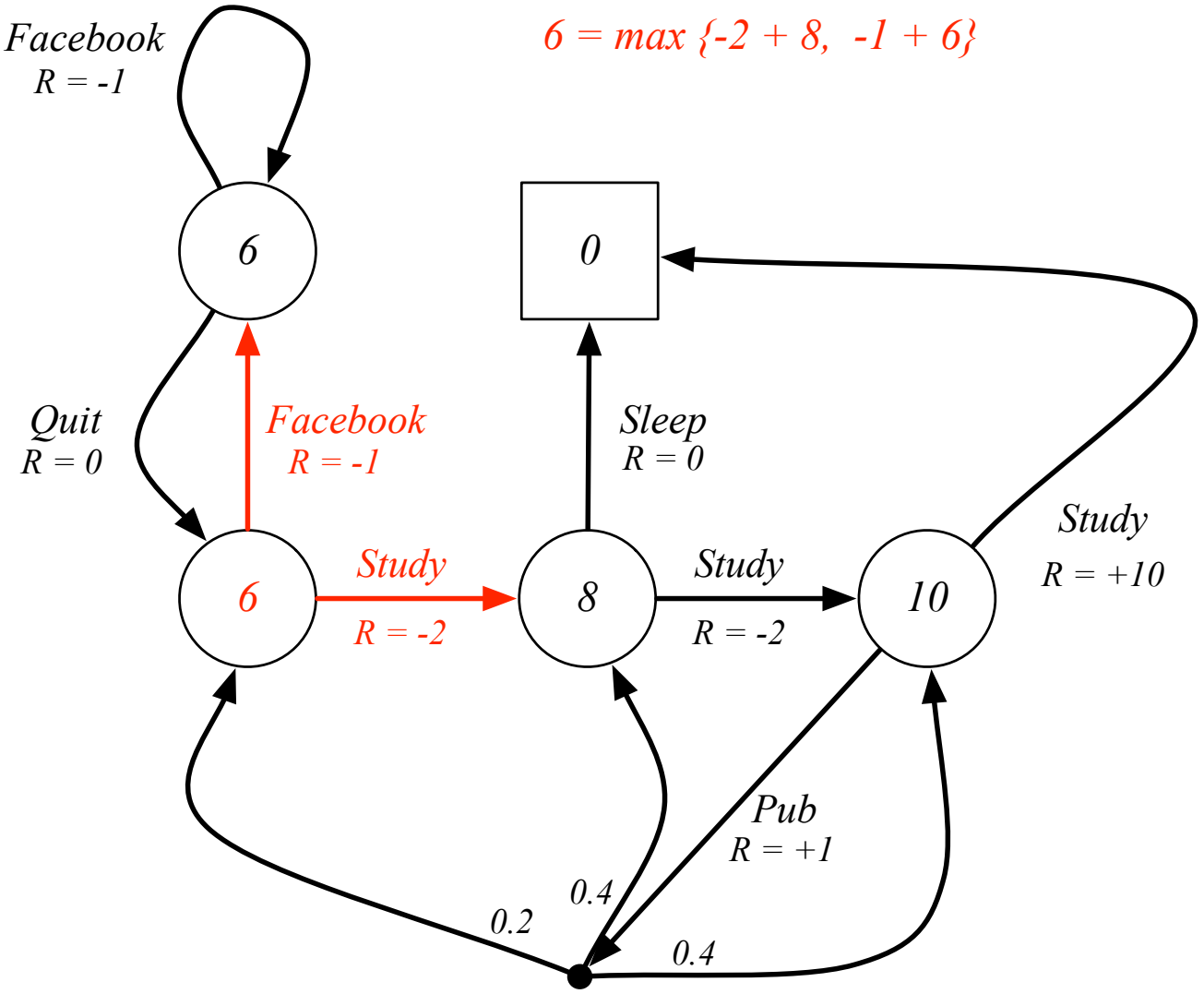
* 이 식은 max 때문에 linear Equation 이 아니다.
따라서 식 정리로 풀 수 없다. (Bellman Expectation Equation 과의 차이점)

Bellman Optimality Equation for Q^* (2)



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

Example: Bellman Optimality Equation in Student MDP



Solving the Bellman Optimality Equation

이것만으로 Optimal 을 구하기 힘들다.
대신에 아래 방법들을 사용한다.

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

Extensions to MDPs (no exam)

- Infinite and continuous MDPs
- Partially observable MDPs
- Undiscounted, average reward MDPs

Infinite MDPs (no exam)

The following extensions are all possible:

- Countably infinite state and/or action spaces
  - Straightforward
- Continuous state and/or action spaces
  - Closed form for linear quadratic model (LQR)
- Continuous time
  - Requires partial differential equations
  - Hamilton-Jacobi-Bellman (HJB) equation
  - Limiting case of Bellman equation as time-step $\rightarrow 0$

POMDPs (no exam)

A Partially Observable Markov Decision Process is an MDP with hidden states. It is a hidden Markov model with actions.

Definition

A *POMDP* is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, \mathcal{R}, \mathcal{Z}, \gamma \rangle$

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

Belief States (no exam)

Definition

A *history* H_t is a sequence of actions, observations and rewards,

$$H_t = A_0, O_1, R_1, ..., A_{t-1}, O_t, R_t$$

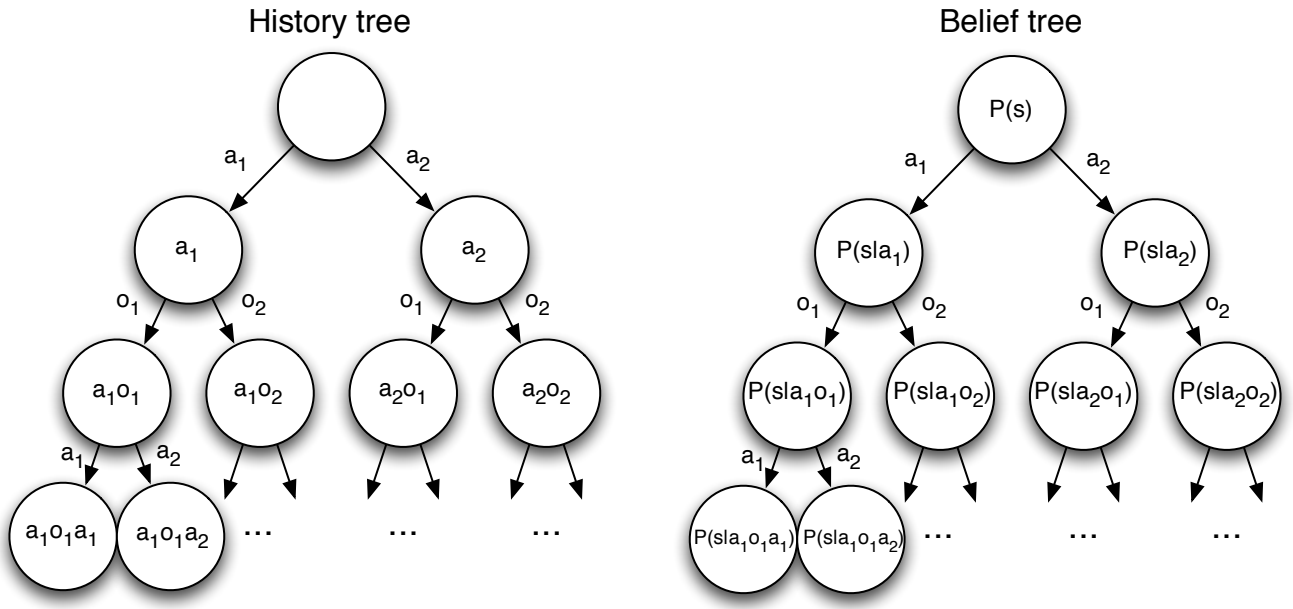
Definition

A *belief state* $b(h)$ is a probability distribution over states, conditioned on the history h

$$b(h) = (\mathbb{P}\left[S_t = s^1 \mid H_t = h\right], ..., \mathbb{P}\left[S_t = s^n \mid H_t = h\right])$$

Reductions of POMDPs (no exam)

- The history H_t satisfies the Markov property
- The belief state $b(H_t)$ satisfies the Markov property



- A POMDP can be reduced to an (infinite) history tree
- A POMDP can be reduced to an (infinite) belief state tree

Ergodic Markov Process (no exam)

An ergodic Markov process is

- *Recurrent*: each state is visited an infinite number of times
- *Aperiodic*: each state is visited without any systematic period

Theorem

An ergodic Markov process has a limiting stationary distribution $d^{\pi}(s)$ *with the property*

$$d^{\pi}(s) = \sum_{s' \in \mathcal{S}} d^{\pi}(s') \mathcal{P}_{s's}$$

Ergodic MDP (no exam)

Definition

An MDP is ergodic if the Markov chain induced by any policy is ergodic.

For any policy π, an ergodic MDP has an *average reward per time-step* ρ^π that is independent of start state.

$$\rho^\pi = \lim_{T \to \infty} \frac{1}{T} \mathbb{E}\left[\sum_{t=1}^{T} R_t\right]$$

Average Reward Value Function (no exam)

- The value function of an undiscounted, ergodic MDP can be expressed in terms of average reward.
- $\tilde{v}_\pi(s)$ is the extra reward due to starting from state s,

$$\tilde{v}_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=1}^{\infty} (R_{t+k} - \rho^\pi) \mid S_t = s \right]$$

There is a corresponding average reward Bellman equation,

$$\begin{aligned} \tilde{v}_\pi(s) &= \mathbb{E}_\pi \left[(R_{t+1} - \rho^\pi) + \sum_{k=1}^{\infty} (R_{t+k+1} - \rho^\pi) \mid S_t = s \right] \\ &= \mathbb{E}_\pi \left[(R_{t+1} - \rho^\pi) + \tilde{v}_\pi(S_{t+1}) \mid S_t = s \right] \end{aligned}$$

Questions?

The only stupid question is the one you were afraid to ask but never did.
-Rich Sutton