

# Lecture 7: Markov Decision Process

Ziyu Shao

School of Information Science and Technology  
ShanghaiTech University

May 08 & 10, 2021

## Outline

- 1 Introduction
- 2 Markov Reward Process
- 3 Markov Decision Process
- 4 References

# Outline

- 1 Introduction
- 2 Markov Reward Process
- 3 Markov Decision Process
- 4 References

## Markov Decision Process

- Markov decision processes formally describe an environment for reinforcement learning
- Where the environment is fully observable
- i.e. The current state completely characterizes the process
- Almost all RL problems can be formalized 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

## Definition

A state  $S_t$  is Markovian if and only if

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

- “The future is independent of the past given the present”
- 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

# Markov Chain

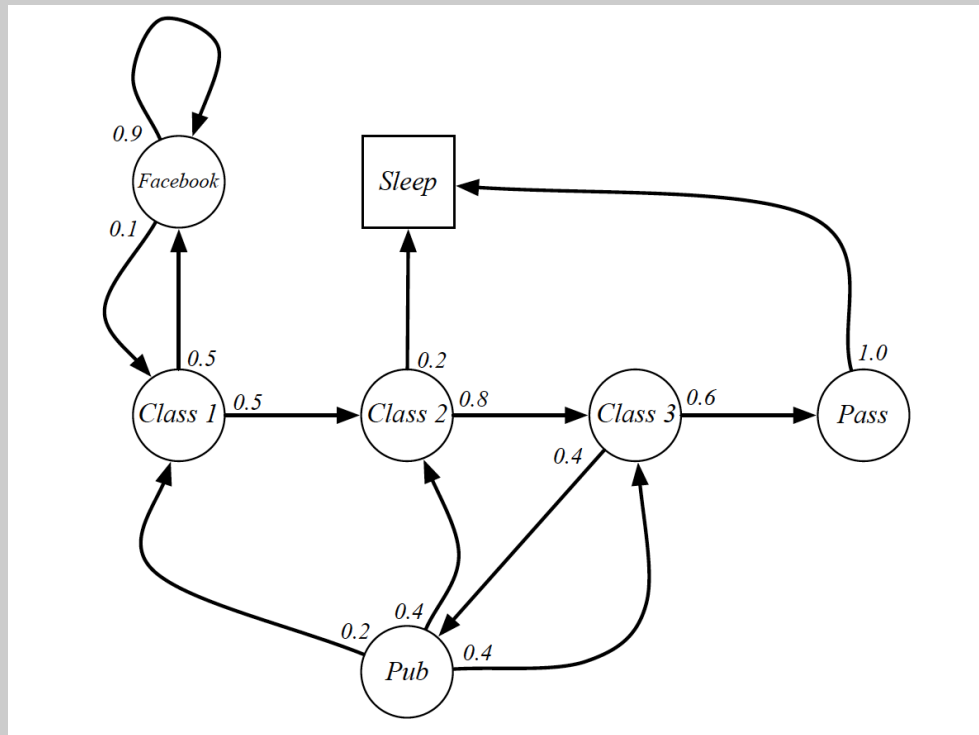
## Definition

A discrete-time Markov chain is a tuple  $\langle \mathcal{S}, \mathcal{P} \rangle$

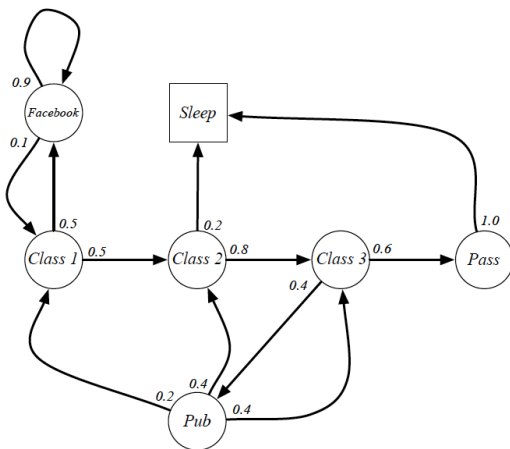
- $\mathcal{S}$  is a (finite) set of states
- $\mathcal{P}$  is a state transition probability matrix

$$\mathcal{P}_{s,s'} = \mathbb{P}[S_{t+1} = s' | S_t = s]$$

## Example: Student Markov Chain



## Example: Student Markov Chain Episodes

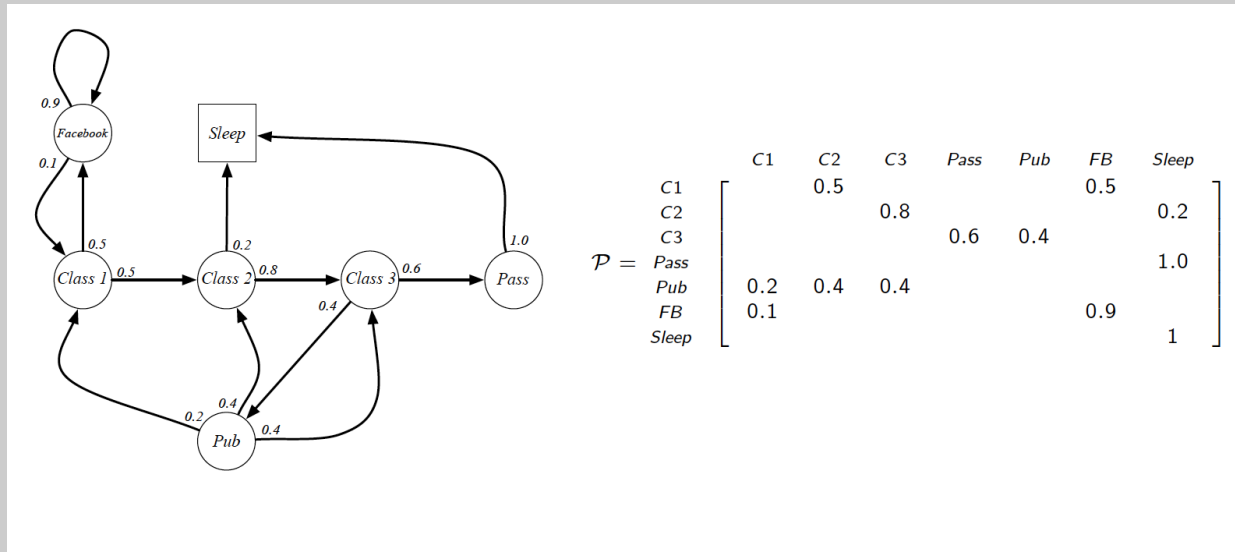


Sample **episodes** for Student Markov Chain starting from  $S_1 = C1$

$S_1, S_2, \dots, S_T$

- 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



## Outline

- 1 Introduction
- 2 Markov Reward Process
- 3 Markov Decision Process
- 4 References

# 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

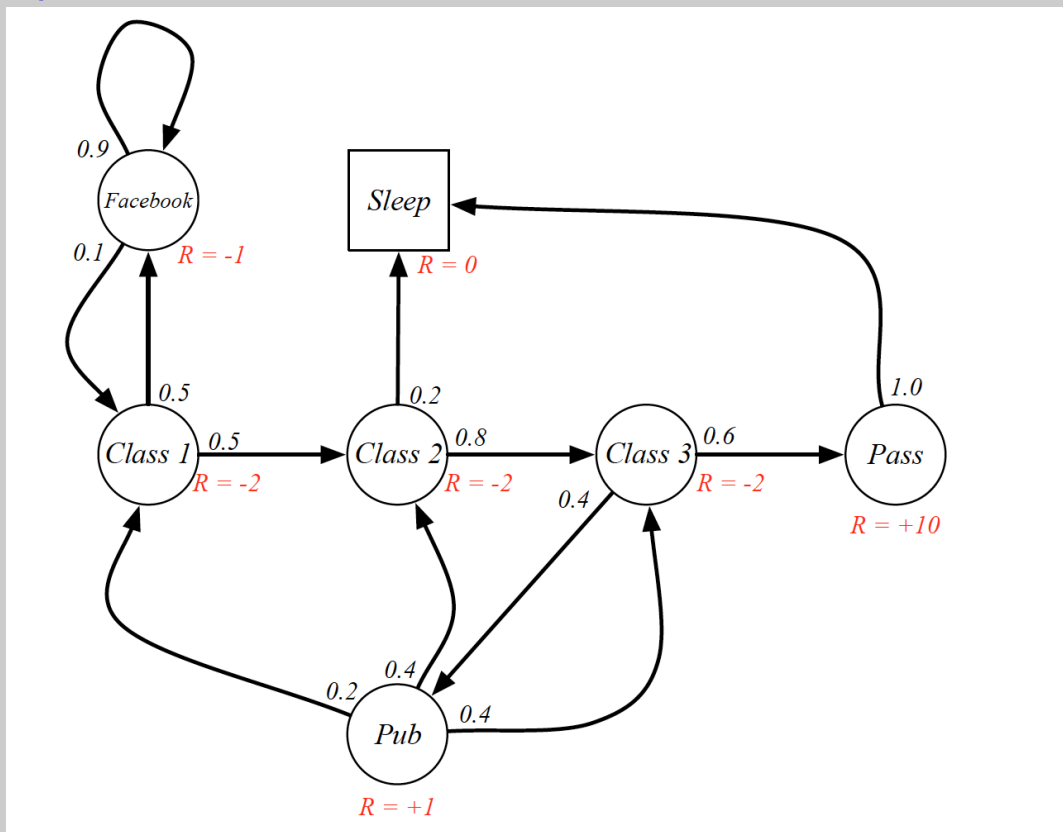
$$\mathcal{P}_{s,s'} = \mathbb{P}[S_{t+1} = s' | S_t = s]$$

- $\mathcal{R}$  is a reward function,

$$\mathcal{R}_s = \mathbb{E}[R_{t+1} | S_t = s]$$

- $\gamma$  is a discount factor,  $\gamma \in [0, 1]$

## Example: Student MRP



# Return

## Definition

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

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

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

## 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 behavior 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.

# 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  $s$

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

## Example: Student MRP Returns

Sample **returns** for Student MRP:

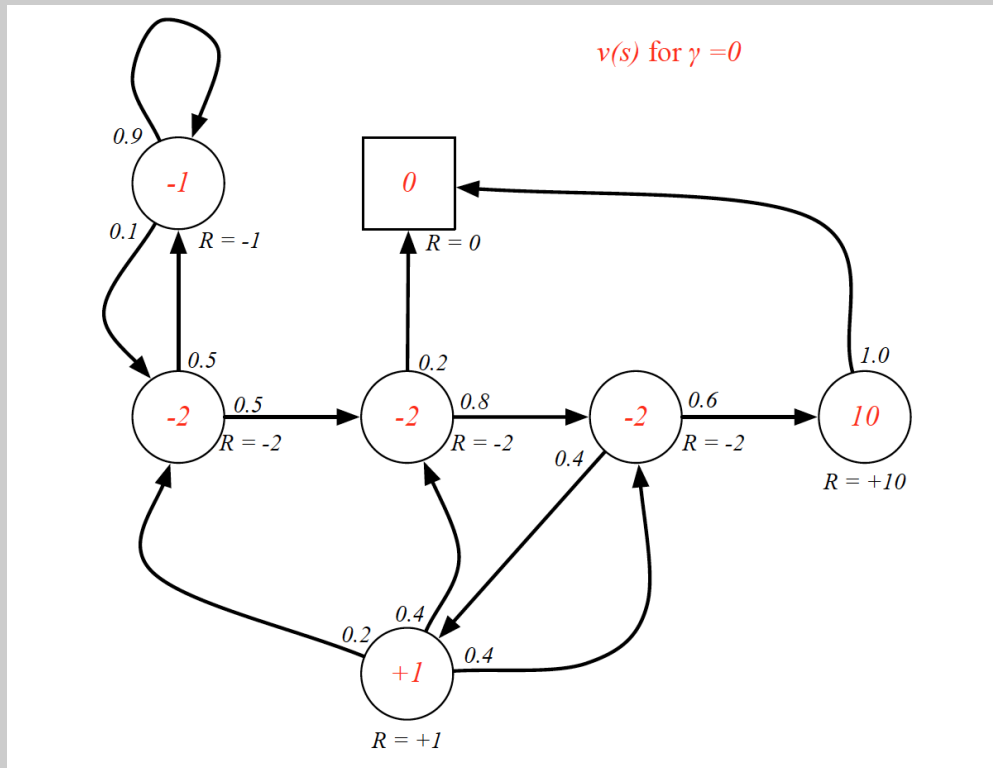
Starting from  $S_1 = C1$  with  $\gamma = \frac{1}{2}$

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

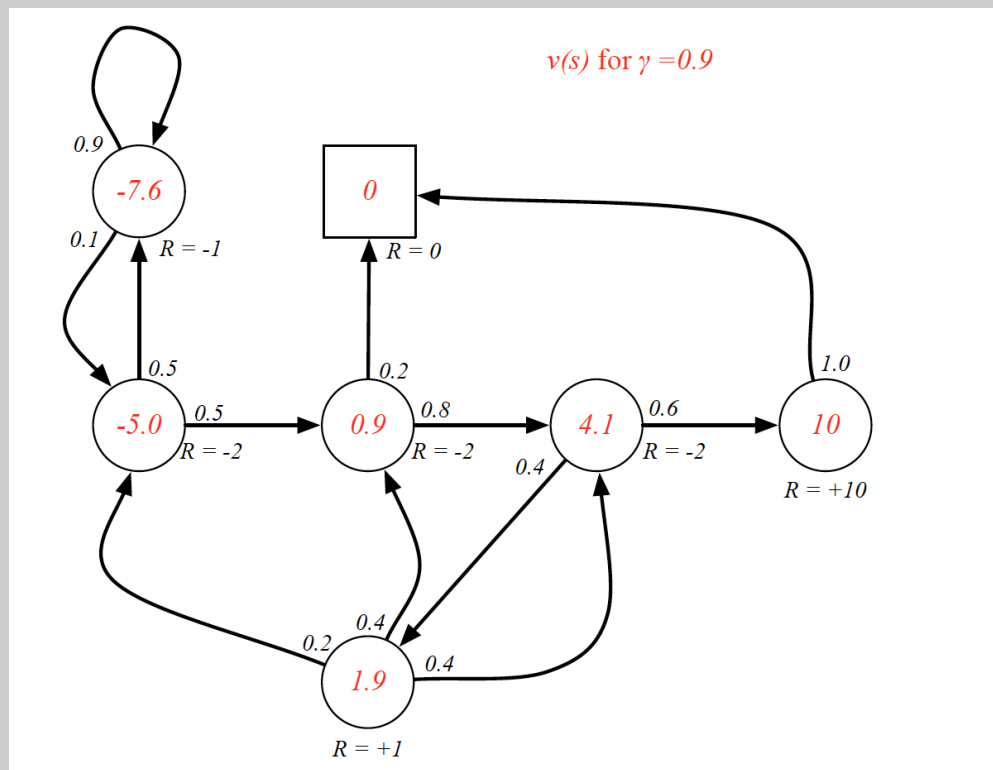
C1 C2 C3 Pass Sleep	$v_1 = -2 - 2 * \frac{1}{2} - 2 * \frac{1}{4} + 10 * \frac{1}{8}$	=	-2.25
C1 FB FB C1 C2 Sleep	$v_1 = -2 - 1 * \frac{1}{2} - 1 * \frac{1}{4} - 2 * \frac{1}{8} - 2 * \frac{1}{16}$	=	-3.125
C1 C2 C3 Pub C2 C3 Pass Sleep	$v_1 = -2 - 2 * \frac{1}{2} - 2 * \frac{1}{4} + 1 * \frac{1}{8}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



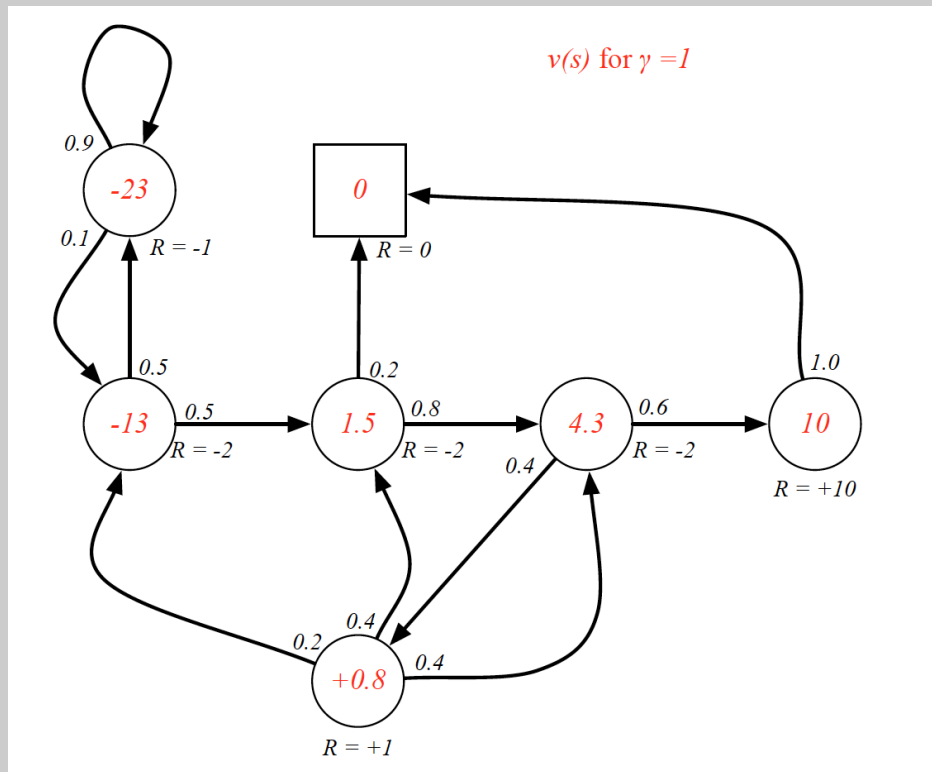
## Example: State-Value Function for Student MRP



## Example: State-Value Function for Student MRP



## Example: State-Value Function for Student MRP



## Bellman Equation for MRPs

The value function can be decomposed into two parts:

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

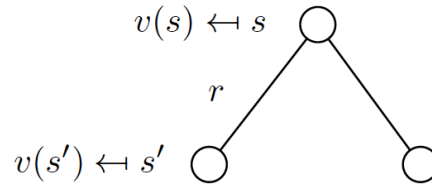
$$\begin{aligned} v(s) &= \mathbb{E}[G_t \mid S_t = s] \\ &= \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s] \\ &= \mathbb{E}[R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \dots) \mid S_t = s] \\ &= \mathbb{E}[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\ &= \mathbb{E}[R_{t+1}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

# Bellman Equation for MRPs

# Bellman Equation for MRPs

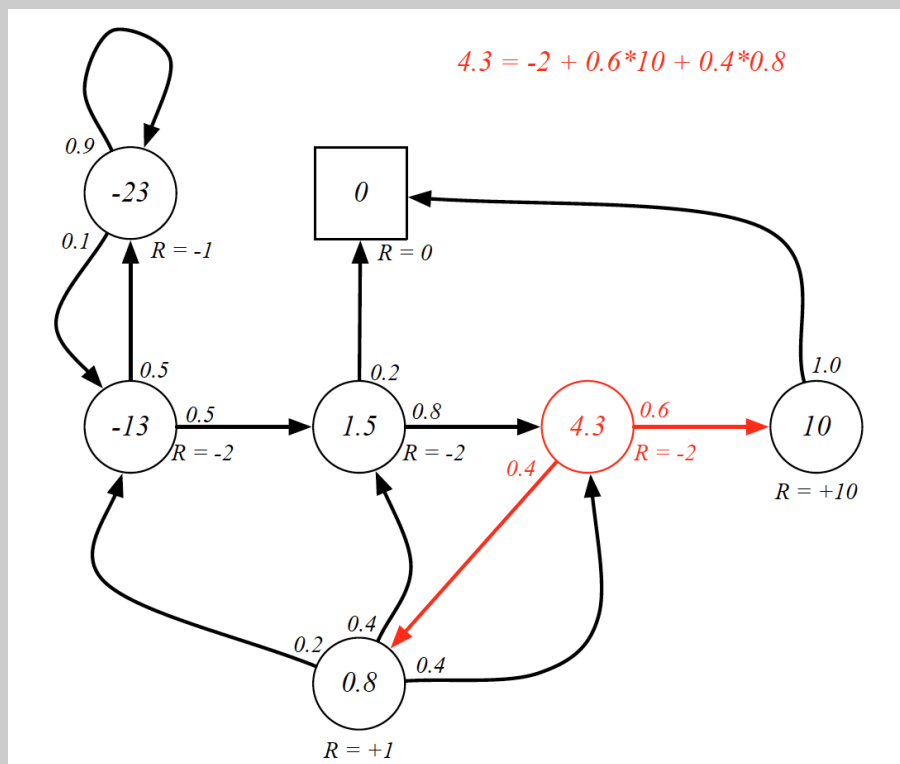
## Bellman Equation for MRP

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



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

## Example: Bellman Equation for Student MRP



## Bellman Equation in Matrix Form

The Bellman equation can be expressed concisely using matrices,

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

where  $\mathbf{v}$  is a column vector with one entry per state

$$\begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix} = \begin{bmatrix} \mathcal{R}_1 \\ \vdots \\ \mathcal{R}_n \end{bmatrix} + \gamma \begin{bmatrix} \mathcal{P}_{11} & \cdots & \mathcal{P}_{1n} \\ \vdots & \ddots & \vdots \\ \mathcal{P}_{n1} & \cdots & \mathcal{P}_{nn} \end{bmatrix} \begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix}$$

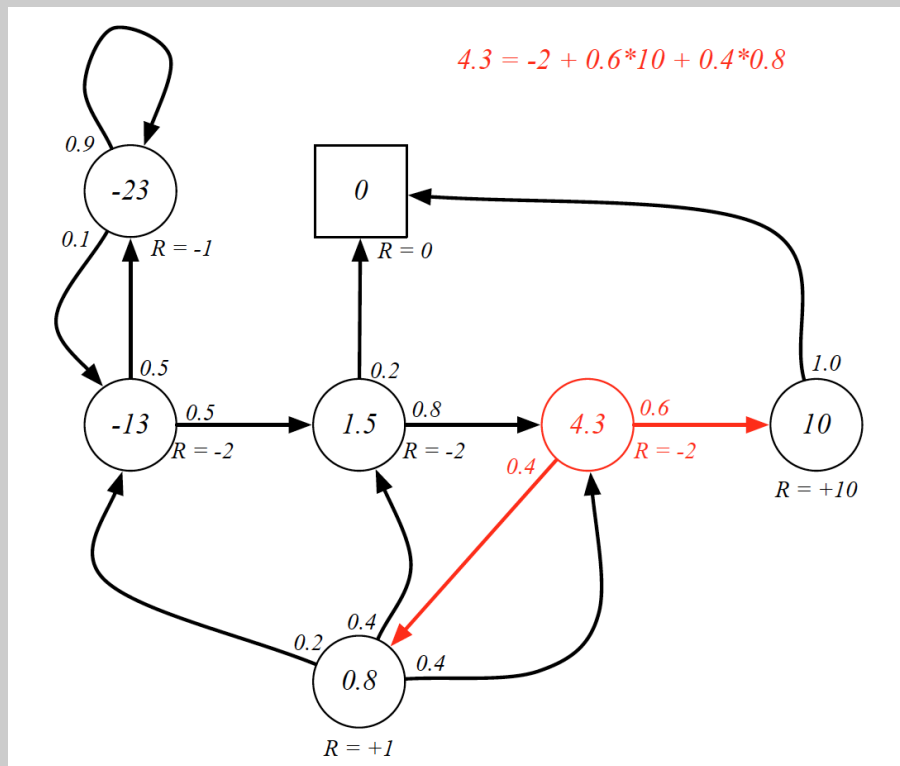
## Solving the Bellman Equation

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

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

- Computational complexity is  $O(n^3)$  for  $n$  states
- Direct solution only possible for small MRPs
- There are many iterative methods for large MRPs, e.g.
  - ▶ Dynamic programming
  - ▶ Monte-Carlo evaluation
  - ▶ Temporal-Different learning

## Example: Matrix Solution for Student MRP



## Example: Matrix Solution for Student MRP

# Example: Matrix Solution for Student MRP

## Outline

- 1 Introduction
- 2 Markov Reward Process
- 3 Markov Decision Process**
- 4 References

# Markov Decision Process

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

## 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

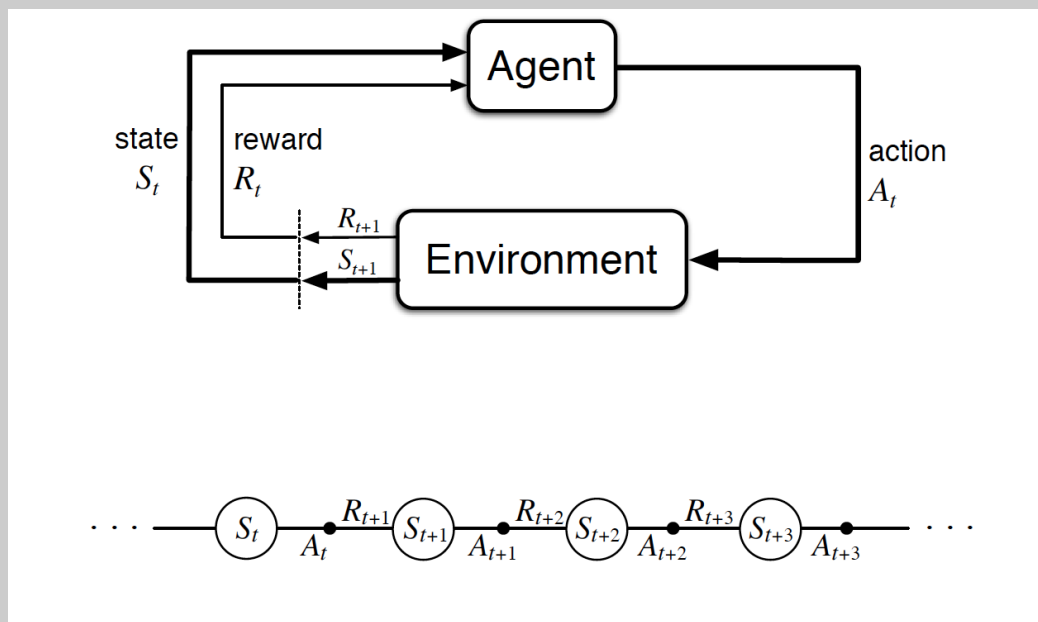
$$\mathcal{P}_{s,s'}^a = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$$

- $\mathcal{R}$  is a reward function,

$$\mathcal{R}_s^a = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$$

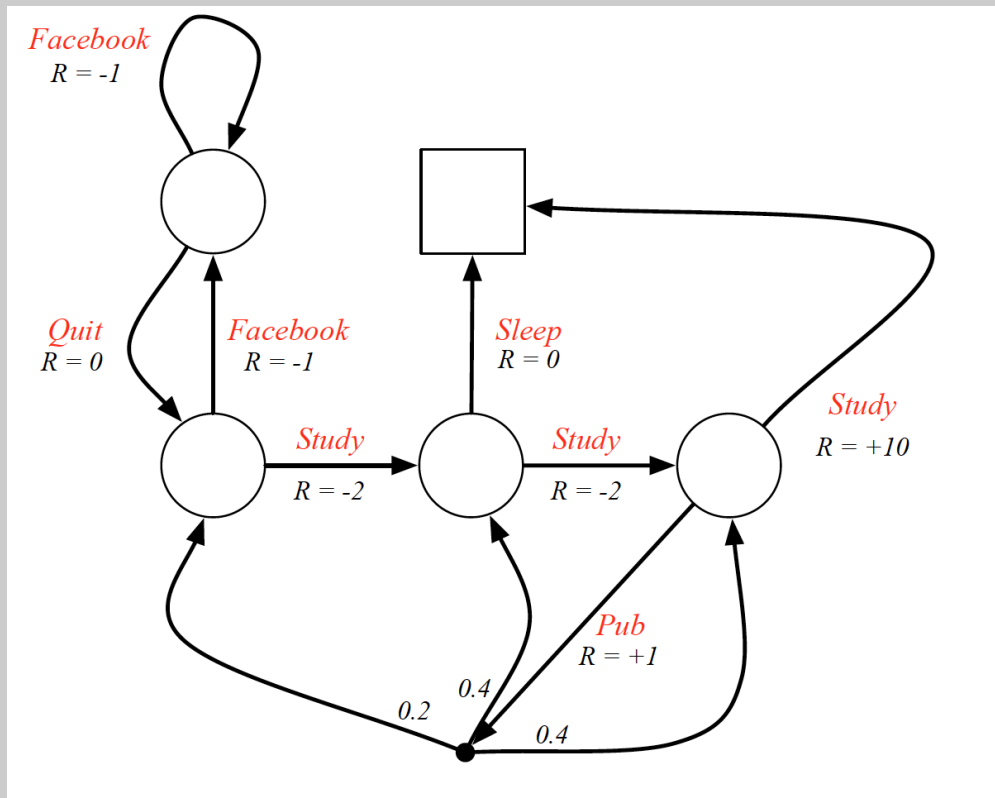
- $\gamma$  is a discount factor,  $\gamma \in [0, 1]$

## The Agent-Environment Interface





## Example: Student MDP



## Policy

### Definition

A *policy*  $\pi$  is a distribution over actions given states,

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

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

## Policy

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

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

## Value Function

### Definition

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

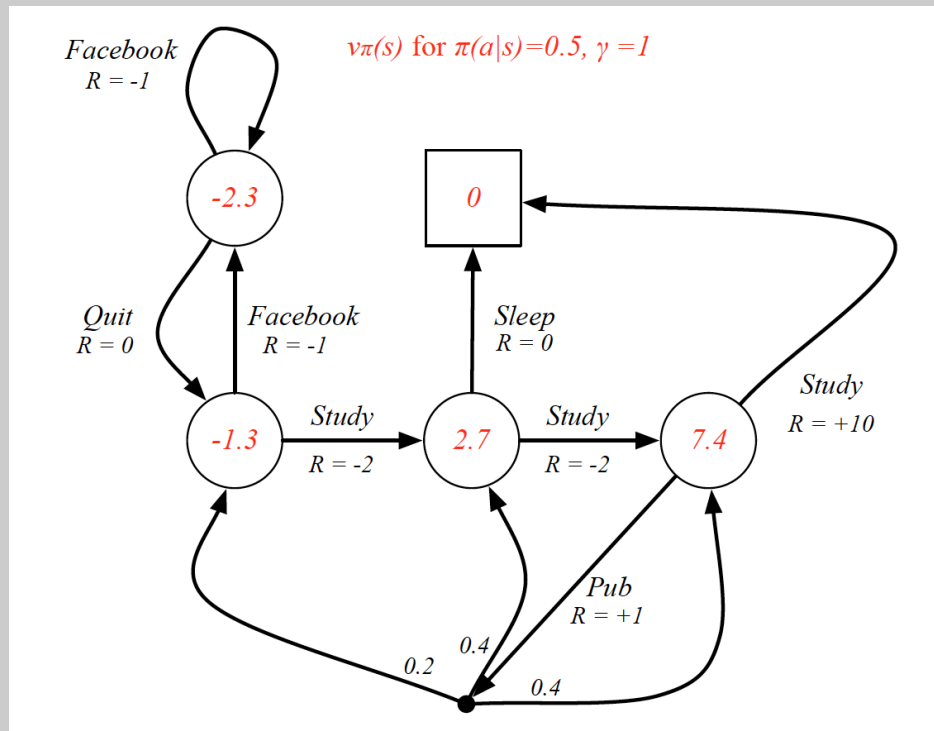
$$v_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$$

### Definition

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

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

## Example: State-Value Function for Student MDP



## Bellman Expectation Equation

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

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

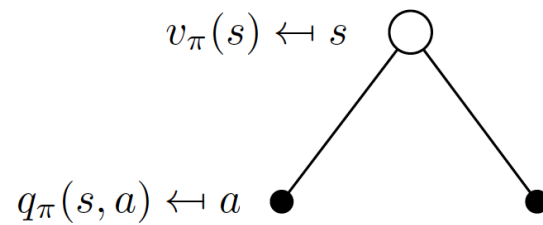
The action-value function can similarly be decomposed,

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

# Bellman Expectation Equation

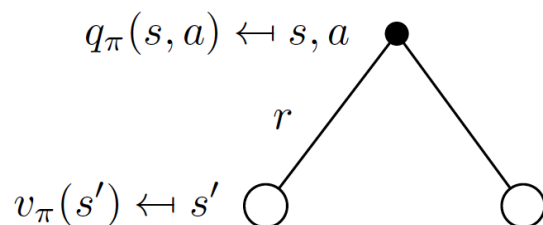
# Bellman Expectation Equation

## Bellman Expectation Equation for $V^\pi$



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

## Bellman Expectation Equation for $Q^\pi$

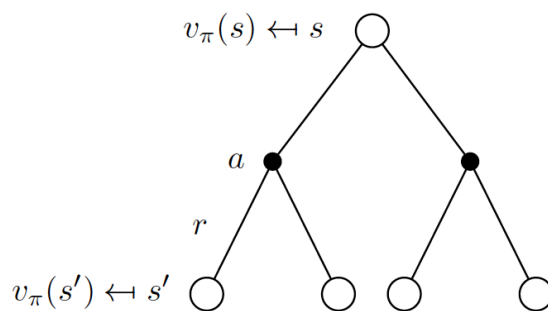


$$q_\pi(s, a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s')$$

# Bellman Expectation Equation for $Q^\pi$

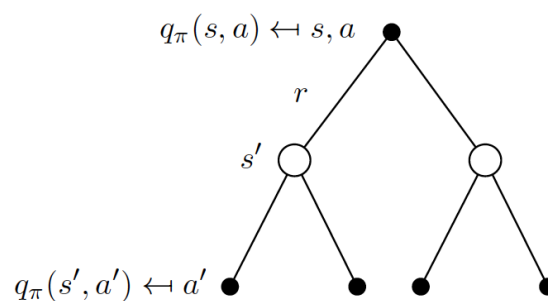
# Bellman Expectation Equation for $Q^\pi$

## Bellman Expectation Equation for $v_\pi$



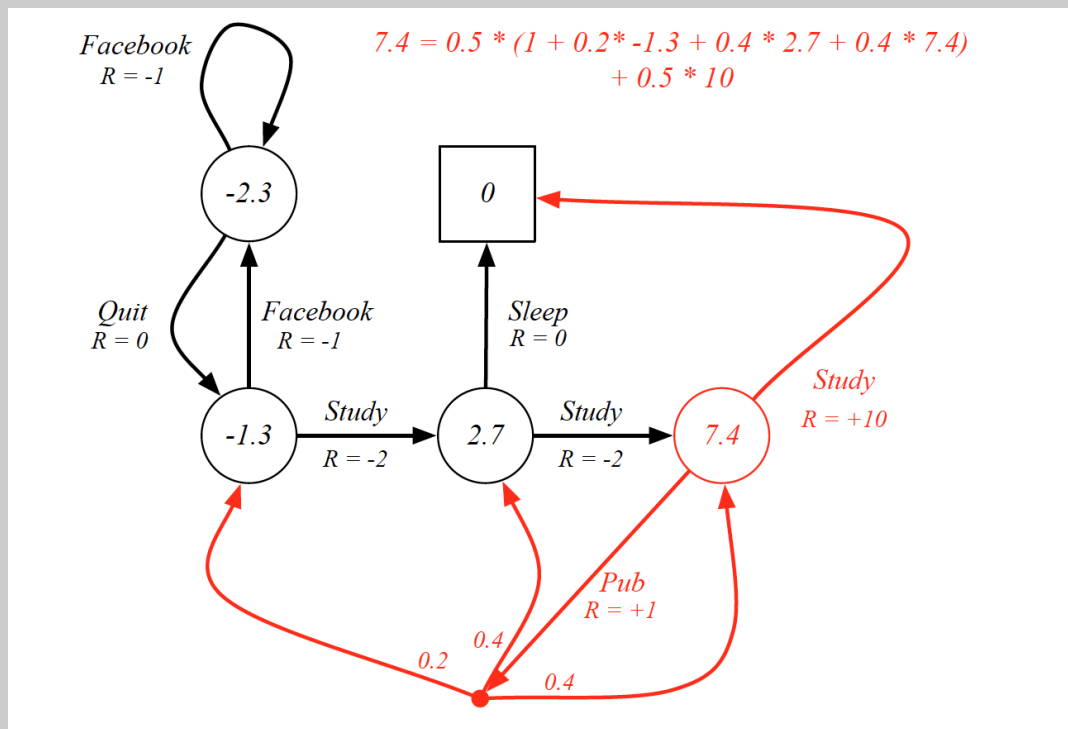
$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left( \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s') \right)$$

## Bellman Expectation Equation for $q_\pi$



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

## Example: Bellman Expectation Equation in Student MDP



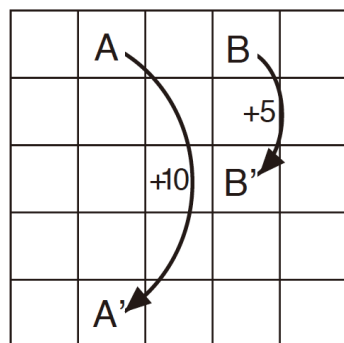
## Example: Bellman Expectation Equation in Student MDP



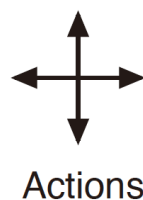
## Example: Bellman Expectation Equation in Student MDP

## Example: Bellman Expectation Equation in Student MDP

## Example: Bellman Expectation Equation in Gridworld



(a)



3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

(b)

What is the value function for the uniform random policy?

## 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

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

## Example: Bellman Expectation Equation (Matrix Form)

## Example: Bellman Expectation Equation (Matrix Form)

# Optimal Value Function

## Definition

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

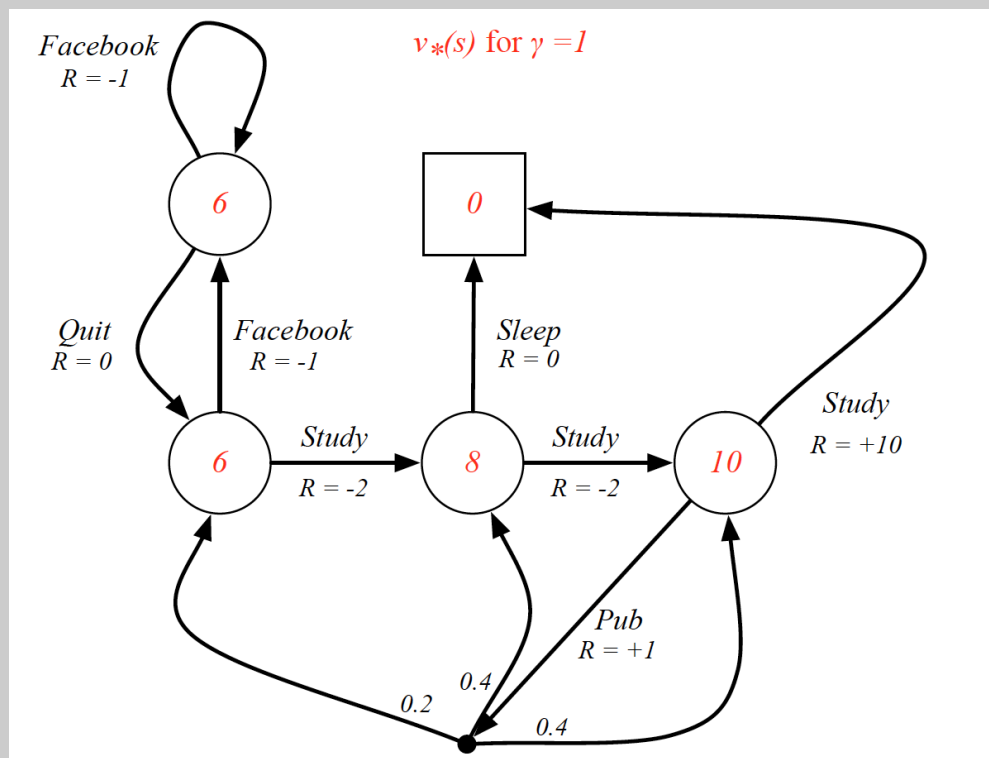
$$v_*(s) = \max_{\pi} v_{\pi}(s)$$

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

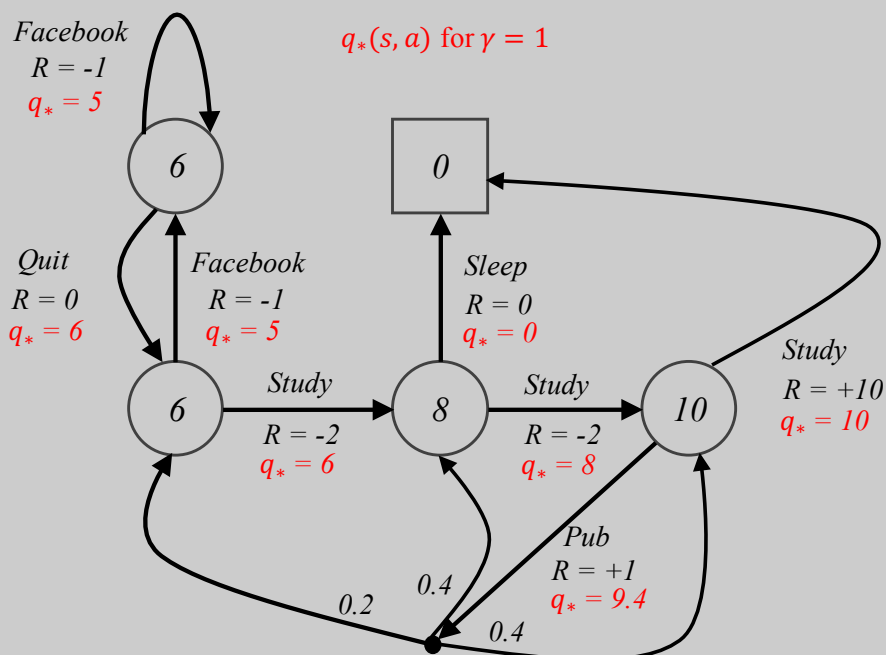
$$q_*(s, a) = \max_{\pi} q_{\pi}(s, a)$$

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

## Example: Optimal Value Function for Student MDP



## Example: Optimal Action-Value Function for Student MDP



## Optimal Policy

Define a partial ordering over policies

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

### Theorem

For any Markov Decision Process

- There exists an optimal policy  $\pi_*$  that is better than or equal to all other policies,  $\pi_* \geq \pi, \forall \pi$
- All optimal policies achieve the optimal value function,  $v_{\pi_*}(s) = v_*(s)$
- All optimal policies achieve the optimal action-value function,  $q_{\pi_*}(s, a) = q_*(s, a)$

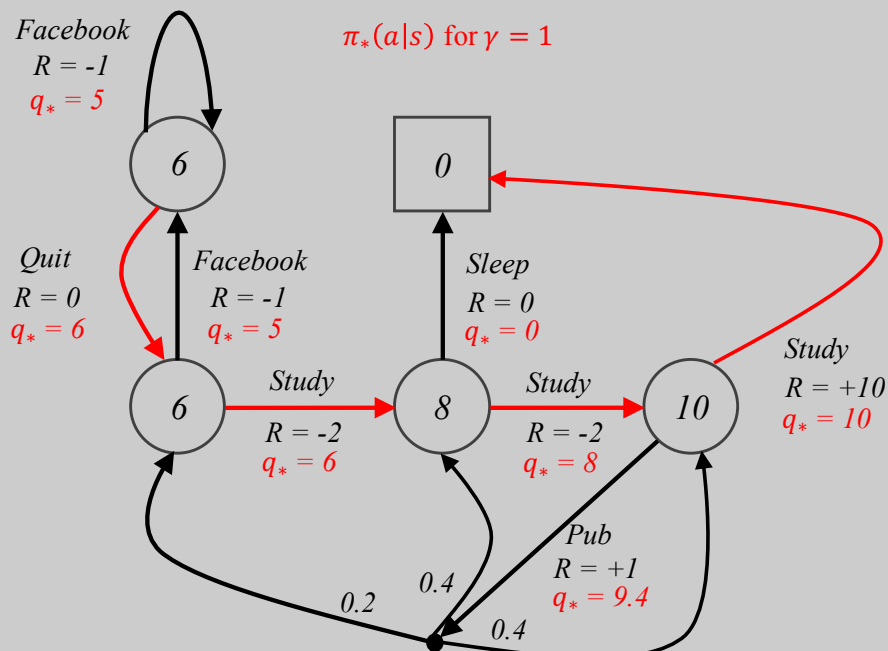
# Finding an Optimal Policy

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

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

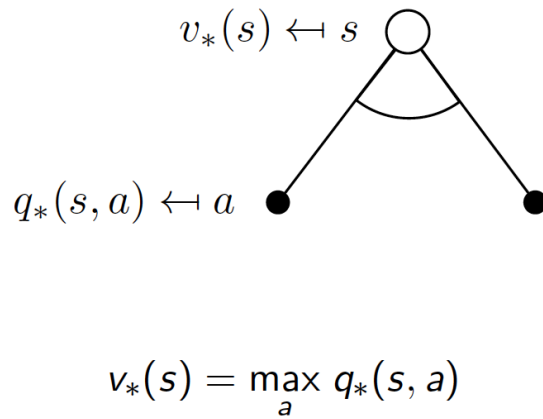
- There is always a deterministic optimal policy for any MDP
- If we know  $q_*(s, a)$ , we immediately have the optimal policy

## Example: Optimal Policy for Student MDP



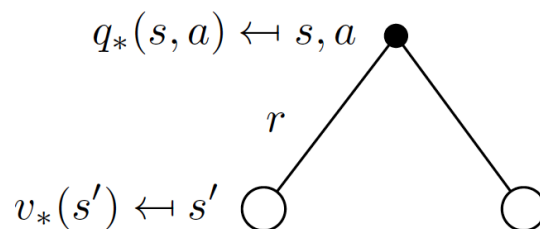
## Bellman Optimality Equation for $v_*$

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



## Bellman Optimality Equation for $Q^*$

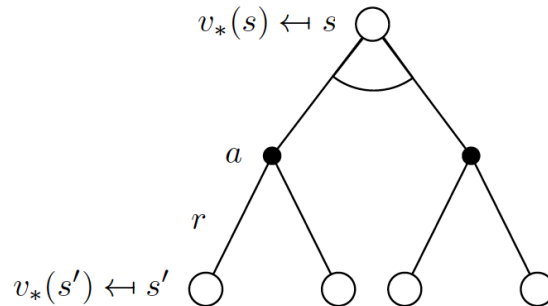
$$q_*(s, a) = \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a]$$



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

## Bellman Optimality Equation for $V^*$

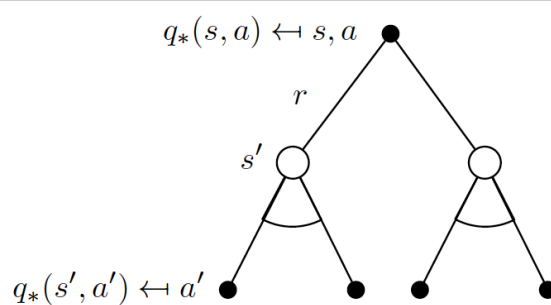
$$v_*(s) = \max_a \mathbb{E} [R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a]$$



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

## Bellman Optimality Equation for $Q^*$

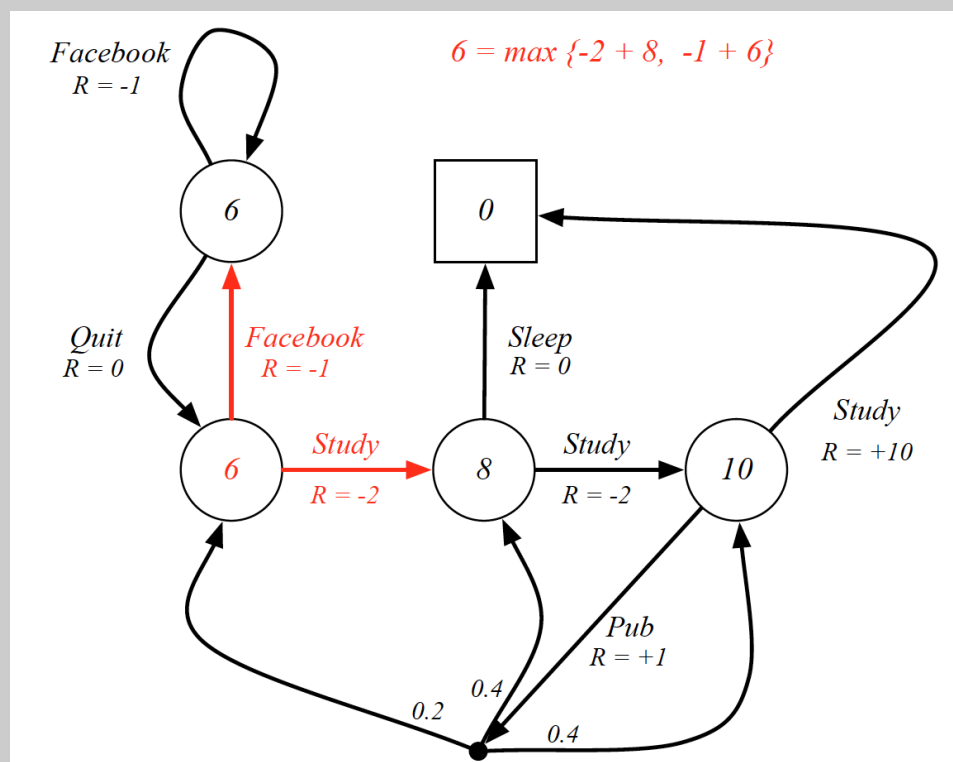
$$q_*(s, a) = \mathbb{E} \left[ R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') | S_t = s, A_t = a \right]$$



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



## Example: Bellman Optimality Equation in Student MDP



## Solution

# Solution

# Solution

# Solution

# Solution

# Solution

# Solution

# Solution

# Solution

## Solving the Bellman Optimality Equation in General

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

# Solving the Bellman Optimality Equation

- Finding an optimal policy by solving the Bellman Optimality Equation requires the following:
  - ▶ accurate knowledge of environment dynamics
  - ▶ we have enough space and time to do the computation
  - ▶ the Markov Property
- How much space and time do we need?
  - ▶ polynomial in number of states
  - ▶ BUT, number of states is often huge
  - ▶ So exhaustive sweeps of the state space are not possible

# Approximation and Reinforcement Learning

- RL methods: Approximating Bellman optimality equations
- Balancing reward accumulation and system identification (model learning) in case of unknown dynamics
- The on-line nature of reinforcement learning makes it possible to approximate optimal policies in ways that put more effort into learning to make good decisions for frequently encountered states, at the expense of less effort for infrequently encountered states.

# Outline

- 1 Introduction
- 2 Markov Reward Process
- 3 Markov Decision Process
- 4 References

## Main References

- Reinforcement Learning: An Introduction (second edition), R. Sutton & A. Barto, 2018.
- RL course slides from Richard Sutton, University of Alberta.
- RL course slides from David Silver, University College London.