


我总算是找到Linux下好用的PDF编辑器啦

Lecture 2: Markov Decision Processes

David Silver

- 1 Markov Processes
- 2 Markov Reward Processes
- 3 Markov Decision Processes
- 4 Extensions to MDPs

Introduction to MDPs

-  *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

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

- 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

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

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

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

状态转移矩阵里面的元素是概率，
所有元素之和是1

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

where each row of the matrix sums to 1.

Markov Process

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

Definition

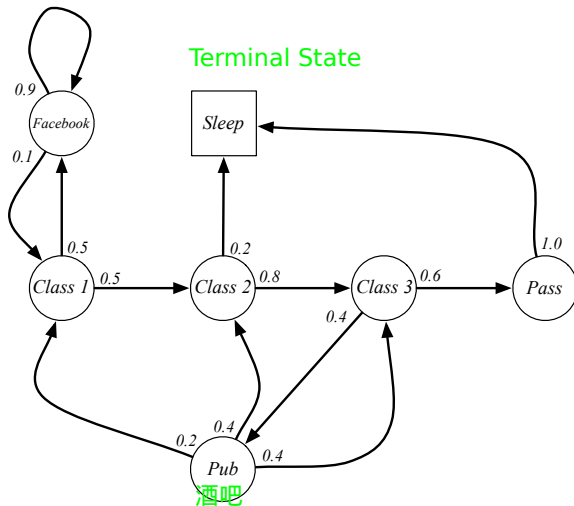
A Markov Process (or Markov Chain) is a tuple $\langle \mathcal{S}, \mathcal{P} \rangle$

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

马尔科夫链==马尔科夫过程,包含两个东西:

1. 状态集 2. 状态转移矩阵

Example: Student Markov Chain

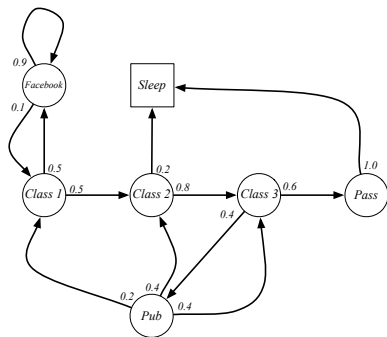


Example: Student Markov Chain Episodes

玩一把游戏就是一个episodes

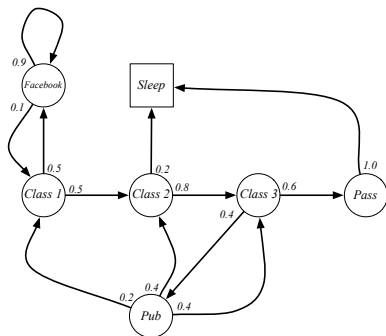
Sample **episodes** for Student Markov Chain starting from $S_1 = C1$

$$S_1, S_2, \dots, S_T$$



- 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



$$\mathcal{P} = \begin{matrix} & \begin{matrix} C1 & C2 & C3 & Pass & Pub & FB & Sleep \end{matrix} \\ \begin{matrix} C1 \\ C2 \\ C3 \\ Pass \\ Pub \\ FB \\ Sleep \end{matrix} & \left[\begin{array}{cccccc} & & 0.5 & & & 0.5 & \\ & & & 0.8 & & & 0.2 \\ & & & & 0.6 & 0.4 & \\ 0.2 & 0.4 & 0.4 & & & & 1.0 \\ 0.1 & & & & & 0.9 & \\ & & & & & & 1 \end{array} \right] \end{matrix}$$

Markov Reward Process

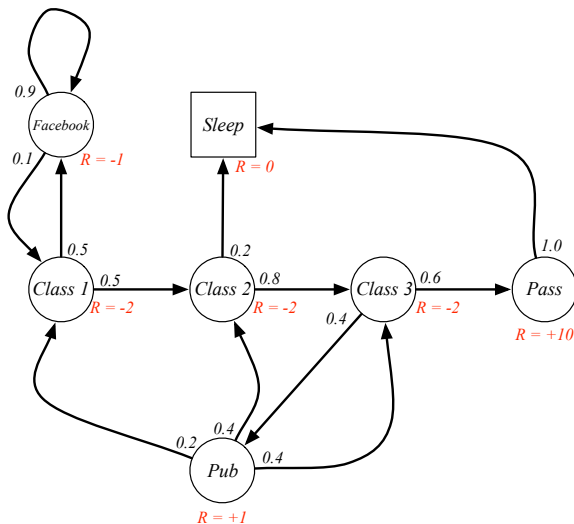
A Markov reward process is a Markov chain with values.

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,
 $\mathcal{P}_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s]$ immediate reward
- \mathcal{R} is a reward function, $\mathcal{R}_s = \mathbb{E}[R_{t+1} \mid S_t = s]$
仅仅在s这个状态下，下一刻我得到的reward是多少
- γ 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 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

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.

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

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


Example: Student MRP Returns

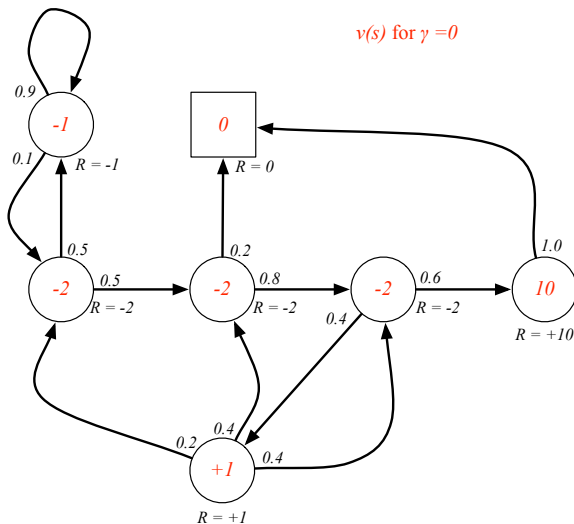
Sample **returns** for Student MRP:

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

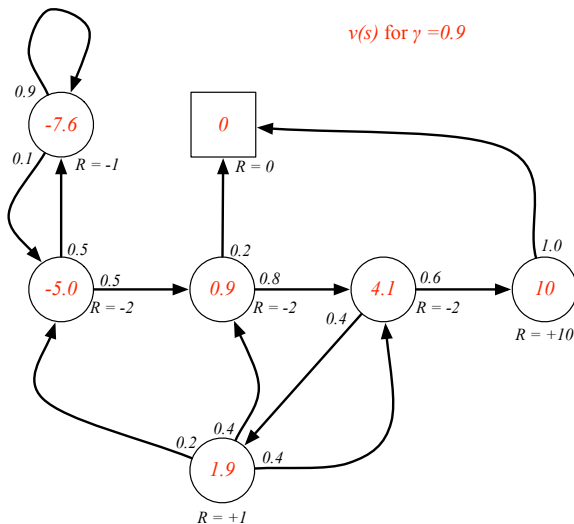
$$G_1 = R_2 + \gamma R_3 + \dots + \gamma^{T-2} R_T$$

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			

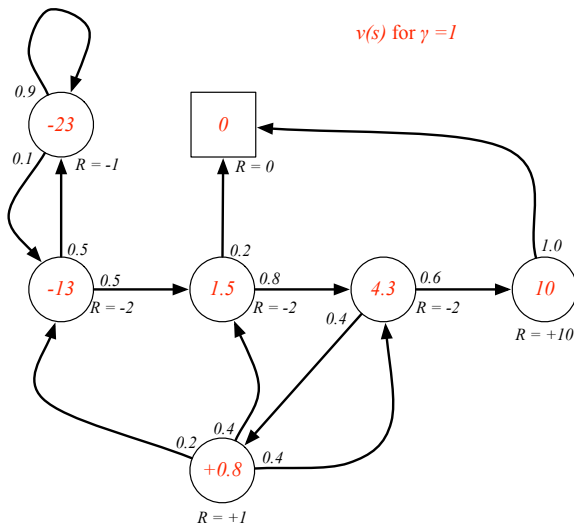
Example: State-Value Function for Student MRP (1)



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



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



Bellman Equation for MRPs

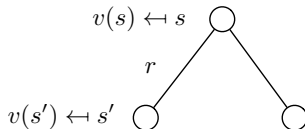
The value function can be decomposed into two parts:

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

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

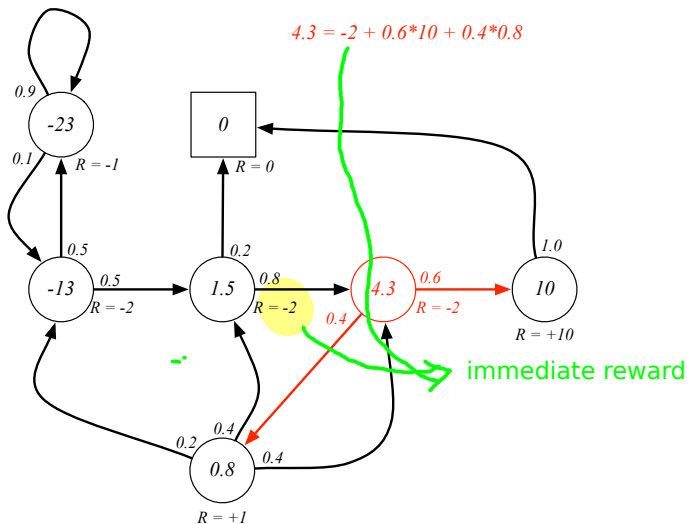
Bellman Equation for MRPs (2)

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



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

Example: Bellman Equation for Student MRP

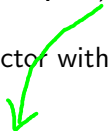


Bellman Equation in Matrix Form

The Bellman equation can be expressed concisely using matrices,

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

where v is a column vector with one entry per state



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

Transition Matrix

总共有n个状态，从状态转移
去理解就好理解了

Solving the Bellman Equation

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

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

$$(I - \gamma \mathcal{P}) v = \mathcal{R}$$

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

计算复杂度是怎么算的?

- 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-Difference learning

这就是为什么要学这些。

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.

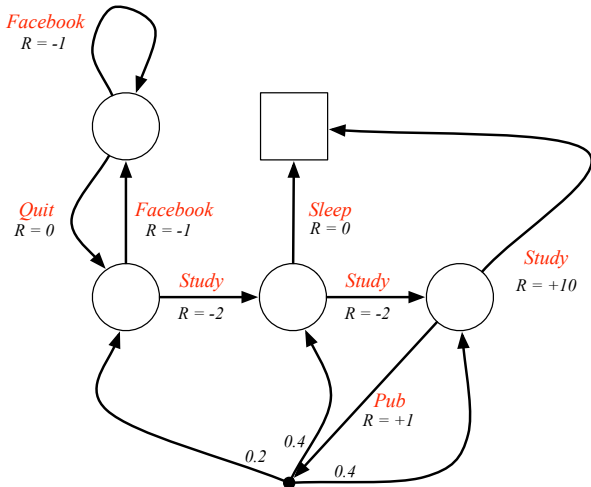
Definition

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 多了一个Action,作为输入
- \mathcal{P} is a state transition probability matrix,
 $\mathcal{P}_{ss'}^a = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$
- \mathcal{R} is a reward function, $\mathcal{R}_s^a = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$
- γ is a discount factor $\gamma \in [0, 1]$.

Example: Student MDP

因为是决策，
这里箭头上就
没有概率可言



去酒吧后的行为是随机的，因此有概率在箭头上

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

和时间无关，只和状态有关

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, \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

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

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

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 π

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

说的是用pi策略，在s这种状态下有多好

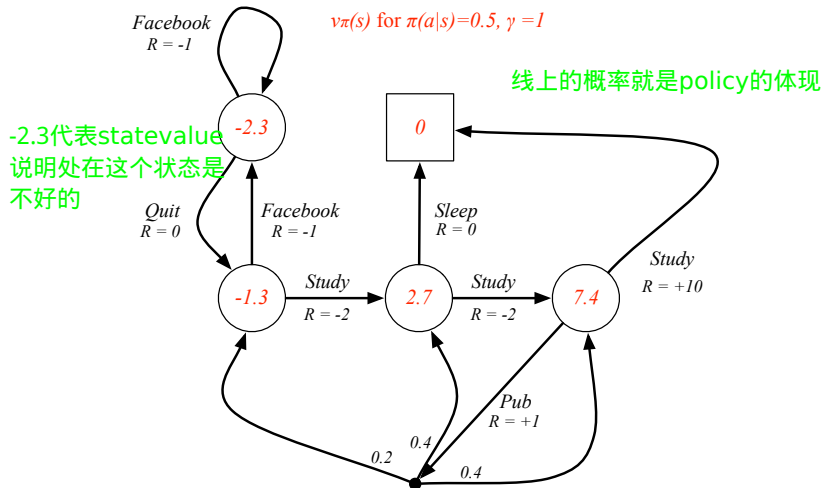
Definition

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

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

说得是用pi策略，在s状态下采取行动a有多好

Example: State-Value Function for Student MDP



Bellman Expectation Equation

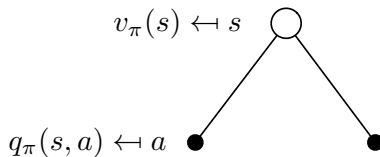
The state-value function can again be decomposed into immediate reward plus discounted value of successor state,

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

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 for V^π



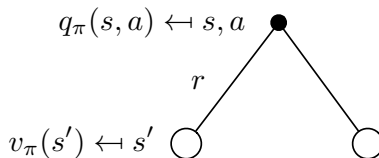
期望

概率*值

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

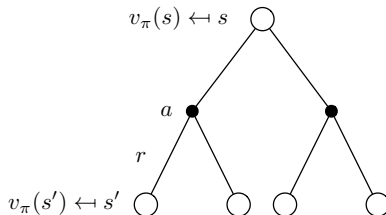
Bellman Expectation Equation for Q^π

关注 action 用黑色
关注 state 用白色



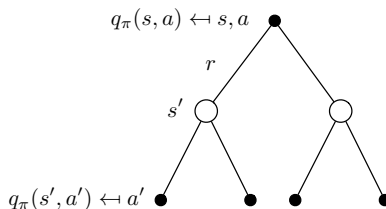
$$q_\pi(s, a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s')$$

Bellman Expectation Equation for v_π (2)



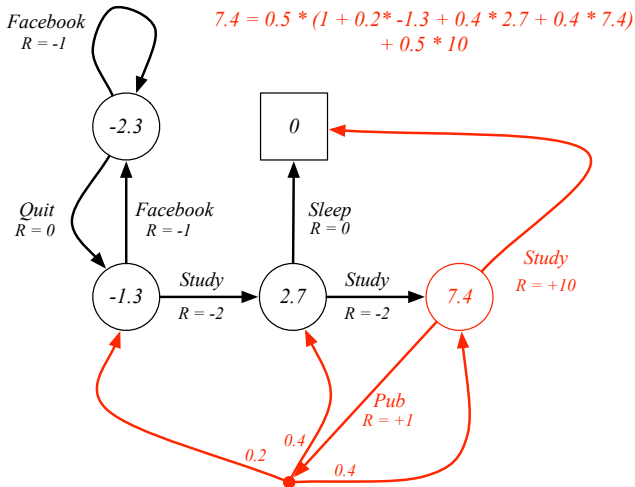
$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s') \right)$$

Bellman Expectation Equation for q_π (2)



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

Example: Bellman Expectation Equation in Student MDP



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

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


Optimal Value Function

Definition

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

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

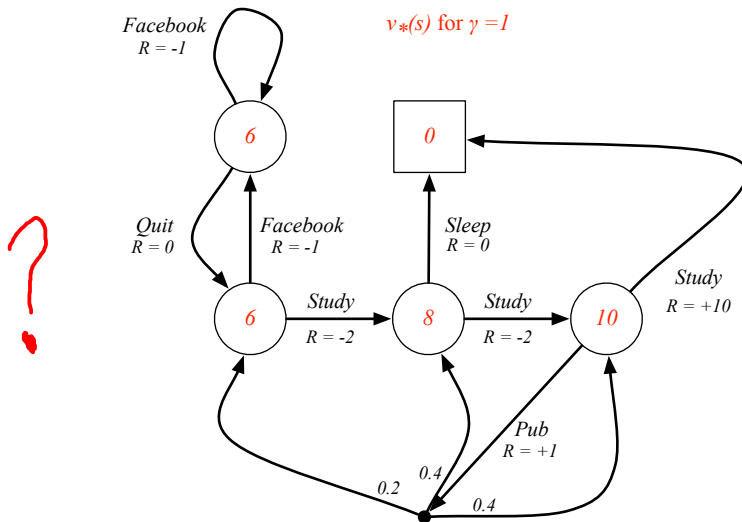
The *optimal action-value function* $q_*(s, a)$ is the maximum action-value function over all policies


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

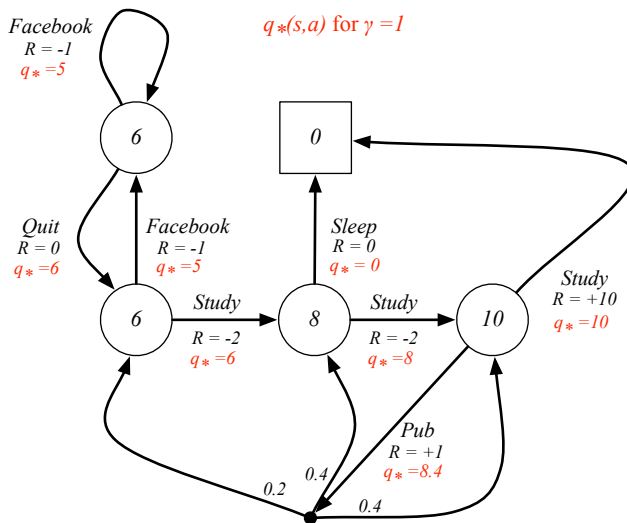
这个是最最终的目的

- 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

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

Theorem

For any Markov Decision Process

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

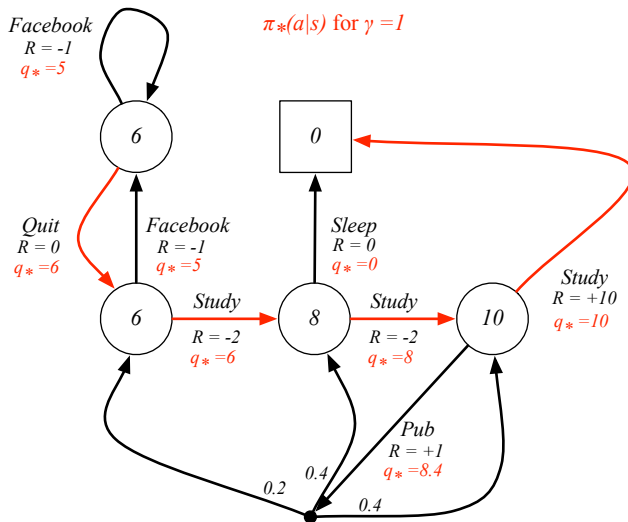
Finding an Optimal Policy

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

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

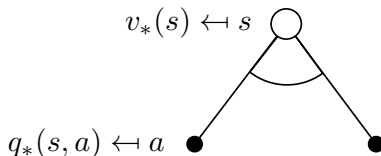
- There is always a deterministic optimal policy for any MDP
- If we know $q_*(s, a)$, we immediately have the optimal policy

Example: Optimal Policy for Student MDP



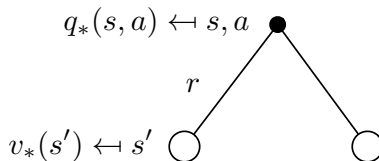
Bellman Optimality Equation for v_*

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



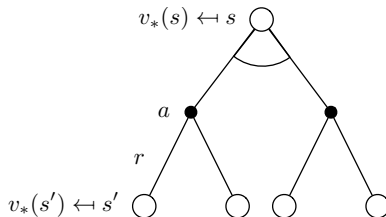
$$v_*(s) = \max_a q_*(s, a)$$

Bellman Optimality Equation for Q^*



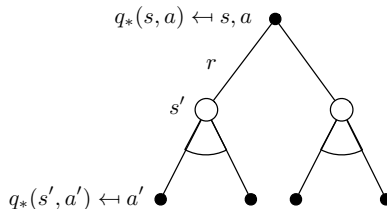
$$q_*(s, a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

Bellman Optimality Equation for V^* (2)



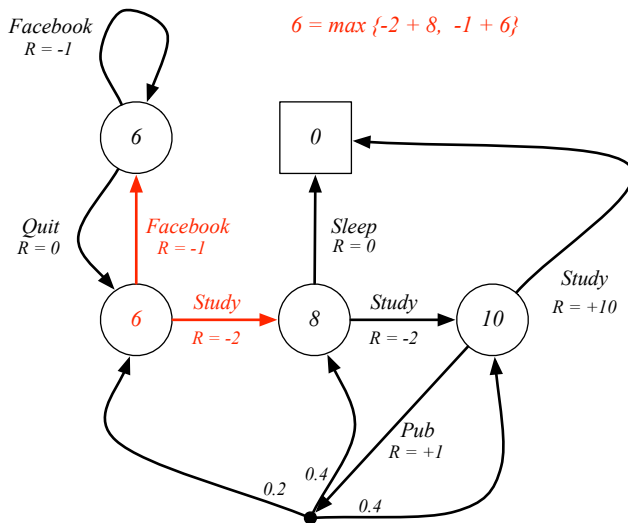
$$v_*(s) = \max_a \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

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

- 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,
 $\mathcal{P}_{ss'}^a = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$
- \mathcal{R} is a reward function, $\mathcal{R}_s^a = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$
- \mathcal{Z} is an observation function,
 $\mathcal{Z}_{s'o}^a = \mathbb{P}[O_{t+1} = o \mid S_{t+1} = s', A_t = a]$
- 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,

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

Definition

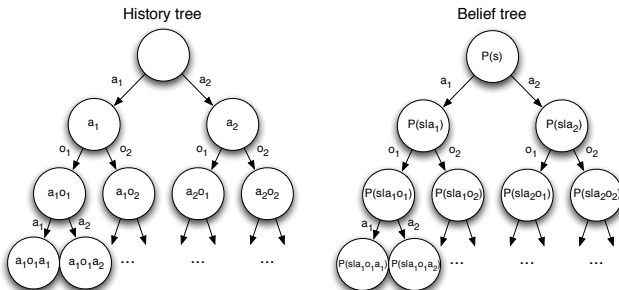
A *belief state* $b(h)$ is a probability distribution over states, conditioned on the history h

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

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.

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

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,

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

Questions?

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

-Rich Sutton