CNRS - Toulouse INP - UT3 - UT Capitole - UT2

Institut de Recherche en Informatique de Toulouse

Lecture 2: MDP

N8EN18B - Contrôle et Apprentissage

Guilherme IECKER RICARDO

IRIT, Université de Toulouse, CNRS, Toulouse INP, UT3

Contents

Markov Processes

Markov Property

"The future is independent of the past give nthe present"

Definition

A state S_t is Markov if, and only if,

$$\mathbb{P}(S_{t+1}|S_t) = \mathbb{P}(S_{t+1}|S_1, \ldots, S_t) \tag{1}$$

- 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

$$\mathcal{P}_{ss'} = \mathbb{P}(S_{t+1} = s' | S_t = s) \tag{2}$$

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

$$\mathcal{P} = \begin{bmatrix} \mathcal{P}_{11} & \dots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{n1} & \dots & P_{nn} \end{bmatrix} \tag{3}$$

where each row of the matrix sums to 1.

Markov Process

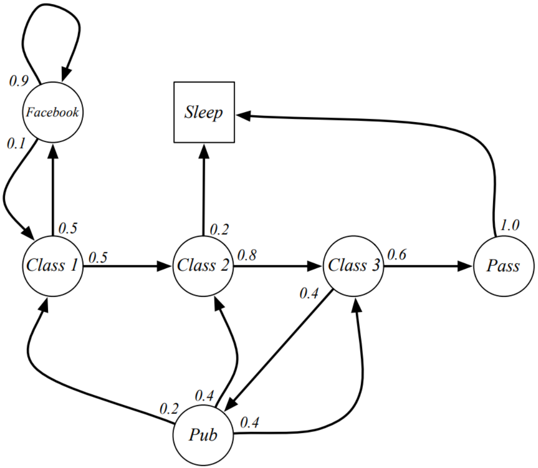
A Markov Process is a “memoryless” random process, i.e., a sequence of random states $S_1, S_2, \ldots$ with the Markov property.

Definition

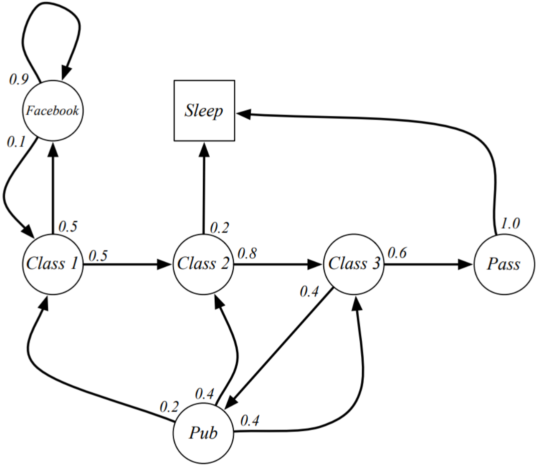
A *Markov Process* (or, simply, a *Markov Chain*) is a tuple $\langle \mathcal{S}, \mathcal{P} \rangle$, where

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

Example: Student Markov Chain



Example: Student Markov Chain's Episodes

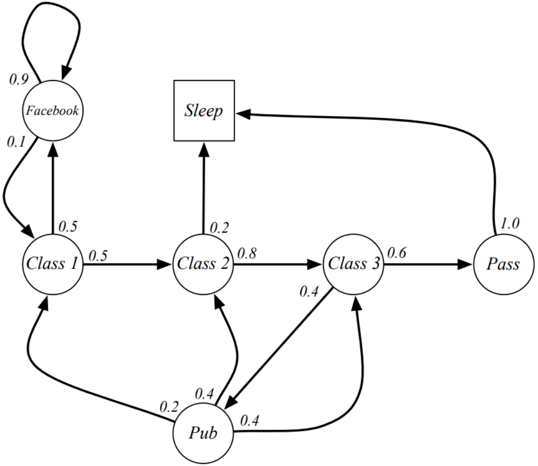


Sample episodes for Student Markov Chain from $S_1 = C1$ with the shape

$$S_1, S_2, \ldots, S_T$$

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

Example: Student MC's Transition Matrix



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

Markov Reward Processes

Markov Reward Process

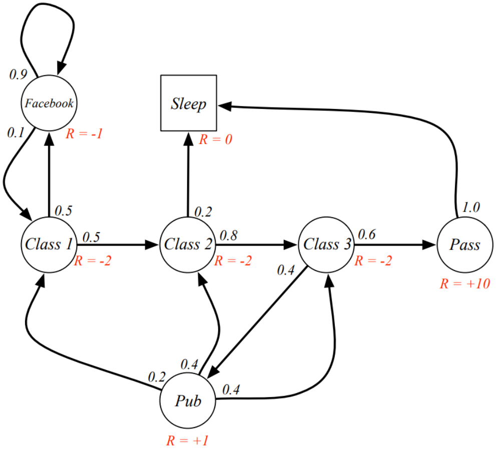
A Markov Reward Process (MRP) is a Markov Process with states' reward values.

Definition

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

- $\mathcal{S}$ is a (finite) set of states
- $\mathcal{P}$ is a state transition probability matrix, $\mathcal{P}_{ss'} = \mathbb{P}(S_{t+1} = s' | S_t = s)$
- $\mathcal{R}$ is a reward function, $\mathcal{R}_s = \mathbb{E}[R_{t+1} | S_t = s]$
- 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, i.e.,

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{4}$$

- 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

State-Value Function

The State-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, i.e.,

$$v(s) = \mathbb{E}[G_t | S_t = s] \tag{5}$$

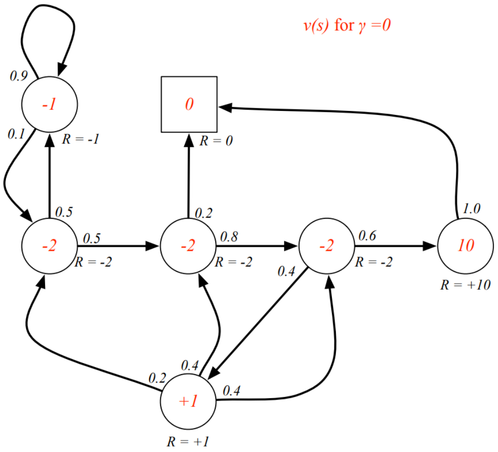
Student MRP: Returns

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

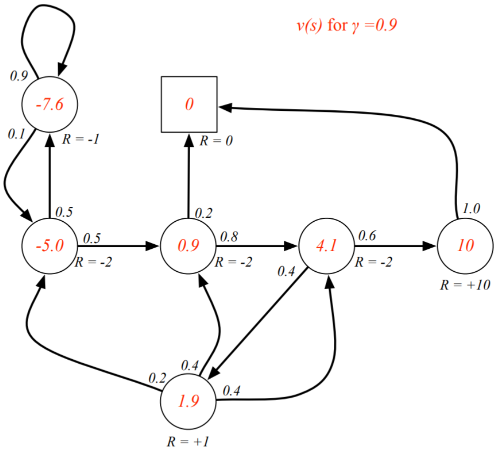
$$G_1 = R_2 + \gamma R_3 + \cdots + \gamma^{T-2} R_T$$

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

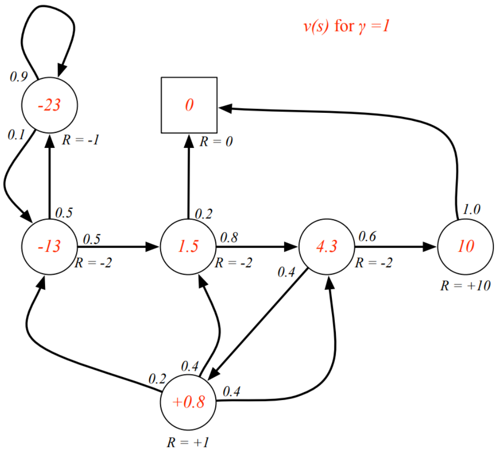
Student MRP: State-Value Function (1)



Student MRP: State-Value Function (2)



Student MRP: State-Value Function (3)



Bellman Equation for MRPs

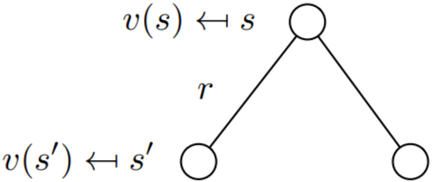
The state-value function can be decomposed into two parts:

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

$$
\begin{aligned}
v(s) &= \mathbb{E}[G_t \,|\, S_t = s] && \text{by def. (5)} \\
&= \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \,|\, S_t = s] && \text{by def. (4)} \\
&= \mathbb{E}[R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \ldots) \,|\, S_t = s] && \text{by distributive} \\
&= \mathbb{E}[R_{t+1} + \gamma G_{t+1} \,|\, S_t = s] && \text{by def. (4)} \\
&= \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \,|\, S_t = s] && \text{by def. (5)}
\end{aligned}
$$

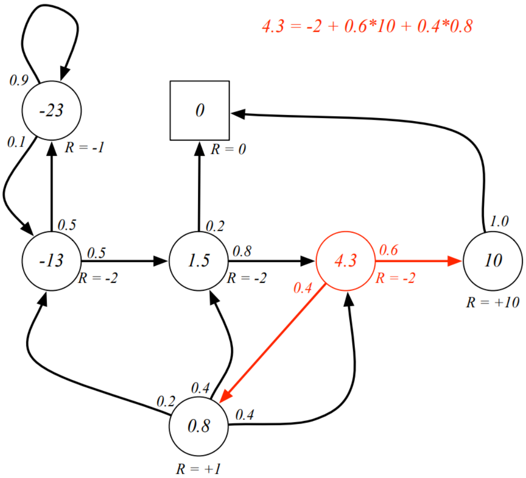
Bellman Equation for MRPs

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



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

Student MRP: Bellman Equation



Bellman Equation in Matrix Form

The Bellman Equation can be expressed concisely using matrices:

$$v = \mathcal{R} + \gamma \mathcal{P} v, \tag{6}$$

where v is a column vector with one entry per state, i.e.,

$$\begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix} = \begin{bmatrix} \mathcal{R}_1 \\ \vdots \\ \mathcal{R}_n \end{bmatrix} + \gamma \begin{bmatrix} \mathcal{P}_{11} & \dots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{n1} & \dots & P_{nn} \end{bmatrix} \begin{bmatrix} v(1) \\ \vdots \\ v(n) \end{bmatrix} \tag{7}$$

Solving the Bellman Equation

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

$$
\begin{aligned}
v &= \mathcal{R} + \gamma \mathcal{P} v \\
(I - \gamma \mathcal{P}) v &= \mathcal{R} \\
v &= (I - \gamma \mathcal{P})^{-1} \mathcal{R}
\end{aligned} \tag{8}
$$

- 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

Markov Decision Processes

Overview

- Markov Decision Processes (MDPs) formally describe an environment for RL
- 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 Decision Process

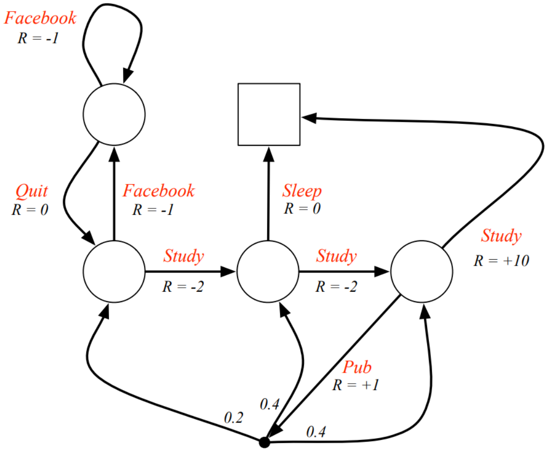
A Markov Decision Process (MDP) is an MRP with decisions. It is an *environment* in which all states are Markov.

Definition

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

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

Student MDP


Facebook
R = -1
Quit
R = 0
Facebook
R = -1
Sleep
R = 0
Study
R = +10
Study
R = -2
Study
R = -2
Pub
R = +1
0.4
0.2
0.4

Policies (1)

Definition

A policy π is a distribution over actions given states,

$$\pi(a|\,s) = \mathbb{P}(A_t = a|\,S_t = s) \tag{9}$$

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

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 $\langle \mathcal{S}, \mathcal{P}^{\pi} \rangle$ and
- the state and reward sequence $S_1, R_1, S_2, R_2, \ldots$ is an MRP $\langle \mathcal{S}, \mathcal{P}^{\pi}, \mathcal{R}^{\pi}, \gamma \rangle$,
- where

$$\begin{aligned} \mathcal{P}^{\pi}_{ss'} &= \sum_{a \in \mathcal{A}} \pi(a|\, s) \mathcal{P}^{a}_{ss'} \\ \mathcal{R}^{\pi}_{s} &= \sum_{a \in \mathcal{A}} \pi(a|\, s) \mathcal{R}^{a}_{s} \end{aligned} \tag{10}$$

Value Functions

Definition

The *State-Value Function* $v_\pi(s)$ of an MDP is the expected return starting from state s, and then following policy π such that

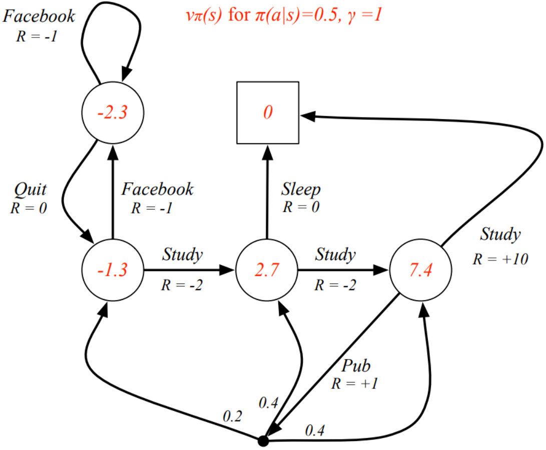
$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] \tag{11}$$

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 such that

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \tag{12}$$

Student MDP: State-Value Function



Bellman Expectation Equation

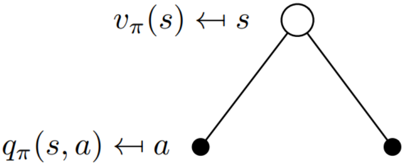
The State-Value Function can again be decomposed into immediate reward plus discounted value of successor state,

$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s] \tag{13}$$

The Action-Value Function can similarly be decomposed,

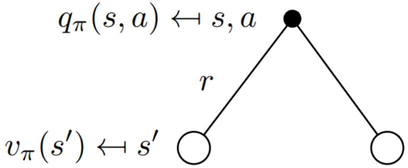
$$q_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | \, S_t = s, A_t = a] \tag{14}$$

Bellman Expectation Equation: Intuition (1/4)



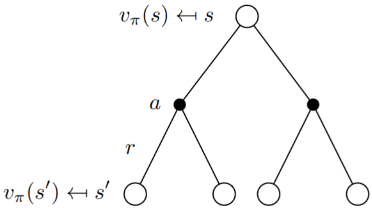
$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) q_\pi(s,a)$$

Bellman Expectation Equation: Intuition (2/4)



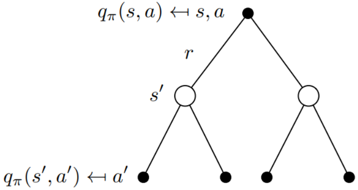
$$q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s')$$

Bellman Expectation Equation: Intuition (3/4)



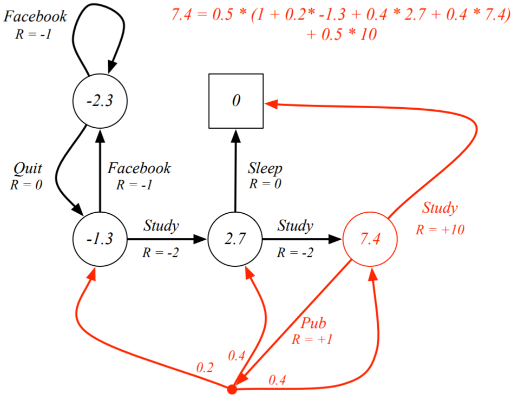
$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s') \right)$$

Bellman Expectation Equation: Intuition (4/4)



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

Student MDP: Bellman Expectation Equation



Bellman Expectation Equation (Matrix Form)

The Bellman Expectation Equation can be expressed concisely using the induced MRP,

$$v_\pi = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi v_\pi \tag{15}$$

with direct solution

$$v_\pi = (I - \gamma \mathcal{P}^\pi)^{-1} \mathcal{R}^\pi \tag{16}$$

Optimal Value Functions

Definition

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

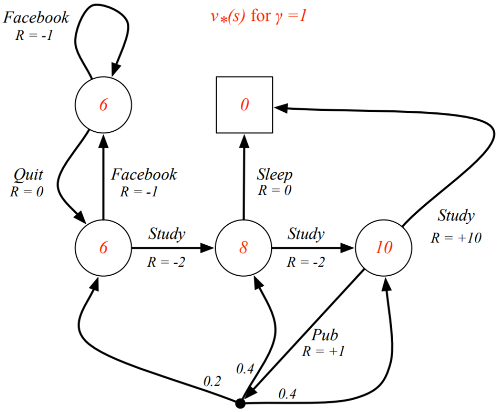
$$v_*(s) = \max_{\pi} v_\pi(s) \tag{17}$$

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

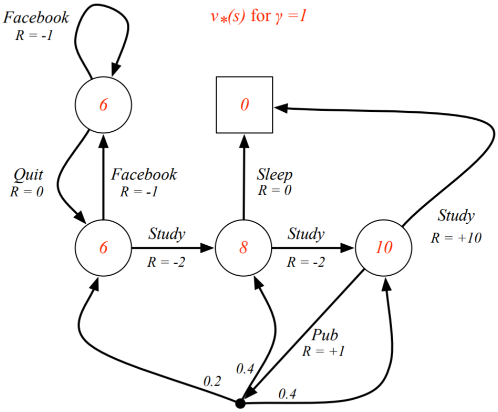
$$q_*(s, a) = \max_{\pi} q_\pi(s, a) \tag{18}$$

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

Student MDP: Optimal State-Value Function



Student MDP: Optimal Action-Value Function



Optimal Policy

Define a partial ordering over policies

$$\pi \geq \pi' \text{ if } v_\pi(s) \geq v_{\pi'}(s), \forall s \tag{19}$$

Theorem

For any MDP:

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

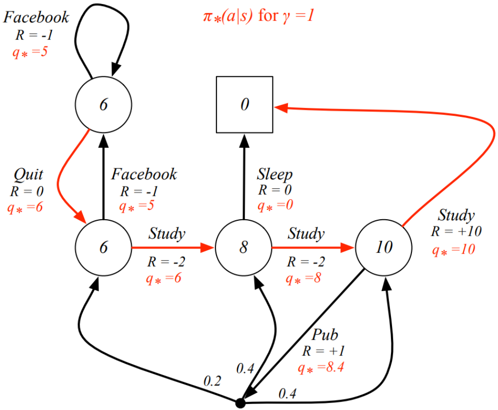
Finding the Optimal Policy

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

$$\pi_*(a|\, s) = \begin{cases} 1, & \text{if } a = \arg\max_{a \in \mathcal{A}} q_*(s,\, a) \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

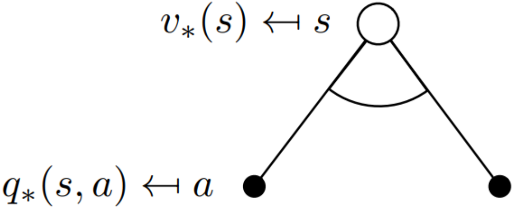
- There is always a deterministic optimal policy for any MDP
- If we know $q_*(s, a)$, we immediately have the optimal policy

Student MDP: Optimal Policy



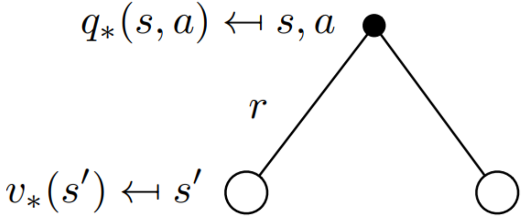
Bellman Optimality Equation for v_*

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



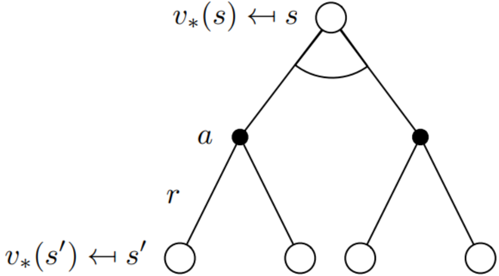
$$v_*(s) = \max_a q_*(s, a)$$

Bellman Optimality Equation for q_*



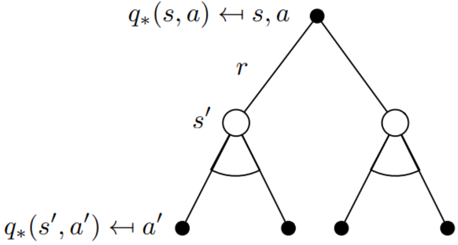
$$q_*(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

Complete Bellman Optimality Equation for v_*



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

Complete Bellman Optimality Equation for q_*



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

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

Expectation vs. Optimality

- Bellman Expectation Equation

$$v_\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|\, s)\left(\mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s')\right)$$
$$q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a \sum_{a'\in\mathcal{A}} \pi(a'|\, s') q_\pi(s',a') \tag{21}$$

- Bellman Optimality Equation

$$v_*(s) = \max_{a\in\mathcal{A}} \mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$
$$q_*(s,a) = \mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a \max_{a'\in\mathcal{A}} q_*(s',a') \tag{22}$$