Lecture 2: Markov Decision Processes

1. Markov Processes 马尔科夫过程

在强化学习中，马尔科夫决策过程（Markov decision process, MDP）是对完全可观测的环境进行描述的，也就是说观测到的状态内容完整地决定了决策需要的特征。几乎所有的强化学习问题都可以用某些方式形式化为一个Markov Decision Process。例如：

- 优化控制问题，比如差分动力学，可以用一个连续的MDP解决
- 部分可观测问题可以转化为MDP
- Bandits：强化学习中探索和开发的困境，这是一种包含state的MDP

1.1 Markov Property 马尔科夫特性

1.1.1 Markov Property

如果一个state具有Markov Property,那就意味着未来的状态独立于过去的状态，它只与现在这一刻的状态有关。也就是说系统的下一个状态S_{t+1}仅与当前状态S_t有关,而与以前的状态无关。

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

Definition

A state S_t is *Markov* if and only if

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

具有Markov Property的state完全特征化了我们需要知道的一切，它从历史数据中捕获了所有的相关信息。一旦知道了当前时刻t,意味着你可以扔掉t时刻之前的拥有Markov Property的state的集合,因为当前状态S_t蕴含了所有相关的历史信息$S_1, S_2, \dots S_{t-1}, S_t$。所以在这样的环境中，一旦当前状态已知，历史信息将会被抛弃。

1.1.2 State Transition Matrix 状态转移矩阵

对于任何拥有Markov Property的问题，对于任何的Markov Process，从某个state开始，可以得到一个后继状态s,这样就可以定义从一个状态转移到另一个状态的概率。公式定义如下：

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

注：状态s转移到状态s'的概率是：当处在时刻t,当前状态为s时，后继的state是随机出现的的某个特定状态s'的概率

我们要时刻记住当前的state特征化了接下来发生的一切，这就意味着一些良好定义的转移概率会告诉我：处于当前的state中，在这种状态下我将以一定的概率值转移到一定的后继状态，这就是状态转移率。

把状态转移矩阵存入矩阵，就得到了状态转移矩阵，状态转移矩阵的每一行都完全特征定制了在MDP中从任何一个可能的开始状态出发进行转移的过程和转移的概率；

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

式中n为状态数量，矩阵中每一行元素之和为1.

1.2 Markov Chains 马尔科夫链

1.2.1 Markov Process Definition 马尔可夫过程的定义

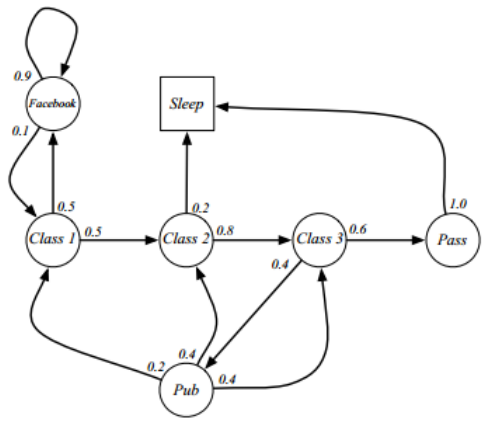
马尔科夫过程又叫马尔科夫链(Markov Chain)，它是一个无记忆的随机过程，是一些具有马尔科夫性质的随机状态序列$S_1, S_2, \dots$构成，可以用一个元组$< S, \mathcal{P} >$表示，其中S是有限数量的状态集，P是状态转移概率矩阵。

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

1.2.2 Example: Student Markov Chain 马尔可夫学生链



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

$S_1, S_2, ..., S_T$

- 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

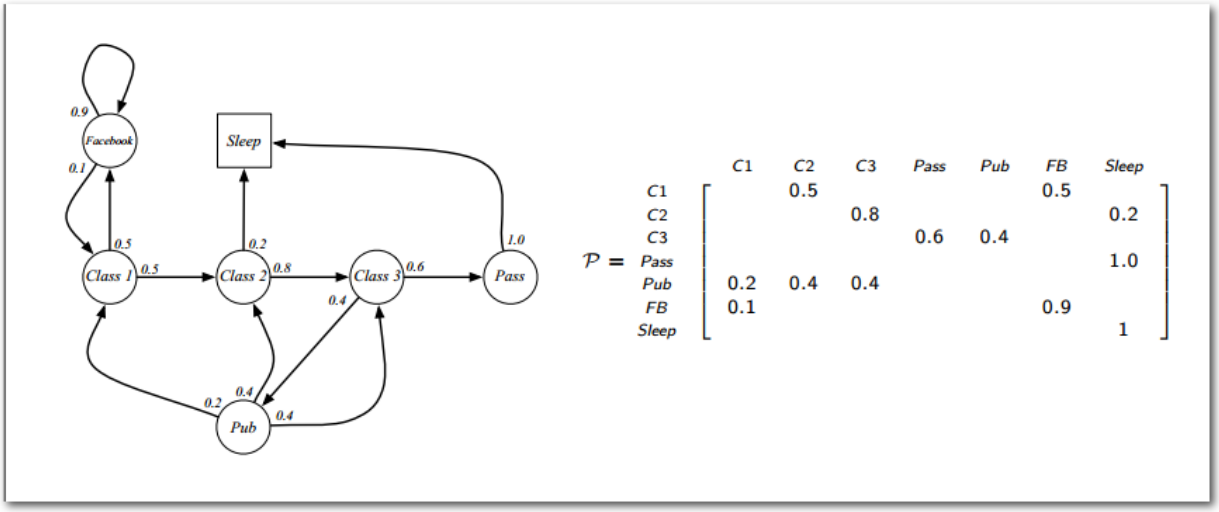
图中圆圈表示学生所处的状态，方格Sleep是一个终止状态，或者可以描述成自循环的状态，也就是Sleep状态的下一个状态100%的几率还是自己。箭头表示状态之间的转移，箭头上的数字表示当前转移的概率。

图片中的序列来自与马尔科夫学生链中的动态采样,这也就意味着从随机的过程中得到一些随机的序列，该序列以一定的概率分布在序列上进行状态转移，这个序列具有Markov Property。

举例说明：当学生处在第一节课（Class1）时，他/她有50%的几率会参加第2节课（Class2）；同时在也有50%的几率不在认真听课，进入到浏览facebook这个状态中。在浏览facebook这个状态时，他/她有90%的几率在下一时刻继续浏览，也有10%的几率返回到课堂内容上来。当学生进入到第二节课（Class2）时，会有80%的几率继续参加第三节课（Class3），也有20%的几率觉得课程较难而退出（Sleep）。当学生处于第三节课这个状态时，他有60%的几率通过考试，继而100%的退出该课程，也有40%的可能性需要到去图书馆之类寻找参考文献，此后根据其对课堂内容的理解程度，又分别有20%、40%、40%的几率返回值第一、二、三节课重新继续学习。一个可能的学生马尔科夫链从状态Class1开始，最终结束于Sleep，其间的过程根据状态转化图可以有很多种可能性，这些都称为Sample Episodes。以下四个Episodes都是可能的：

> 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

该学生马尔科夫过程的状态转移矩阵如下图：



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

状态转移矩阵基本告诉我们，对于我们可能位于的任何state,我们会以多大的概率转移到另外的任一状态。它充分的描述了整个系统的动态过程

2. Markov Reward Processes 马尔科夫奖励过程

2.1 MRP

2.1.1 Definition

Markov Reward Process是带有value判断的Markov Process。除了Markov Process中的状态空间 $\mathcal{S}$ 和转移状态$\mathcal{P}$,再添加reward function (奖励函数)$\mathcal{R}$和discount factor(折扣因子)γ。$\mathcal{R}$告诉我们当前时刻的这一步，我们在时间T，状态s的时候，时间T加1，可以获得的reward。衰减因子$\gamma \in [0,1]$,视频中David列举了不少原因来解释为什么引入衰减系数，其中有数学表达的方便，这也是最重要的；避免陷入无限循环；远期利益具有一定的不确定性；符合人类对于眼前利益的追求；符合金融学上立即的回报相对于延迟的汇报能够获得更多的利益等等。

Definition

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

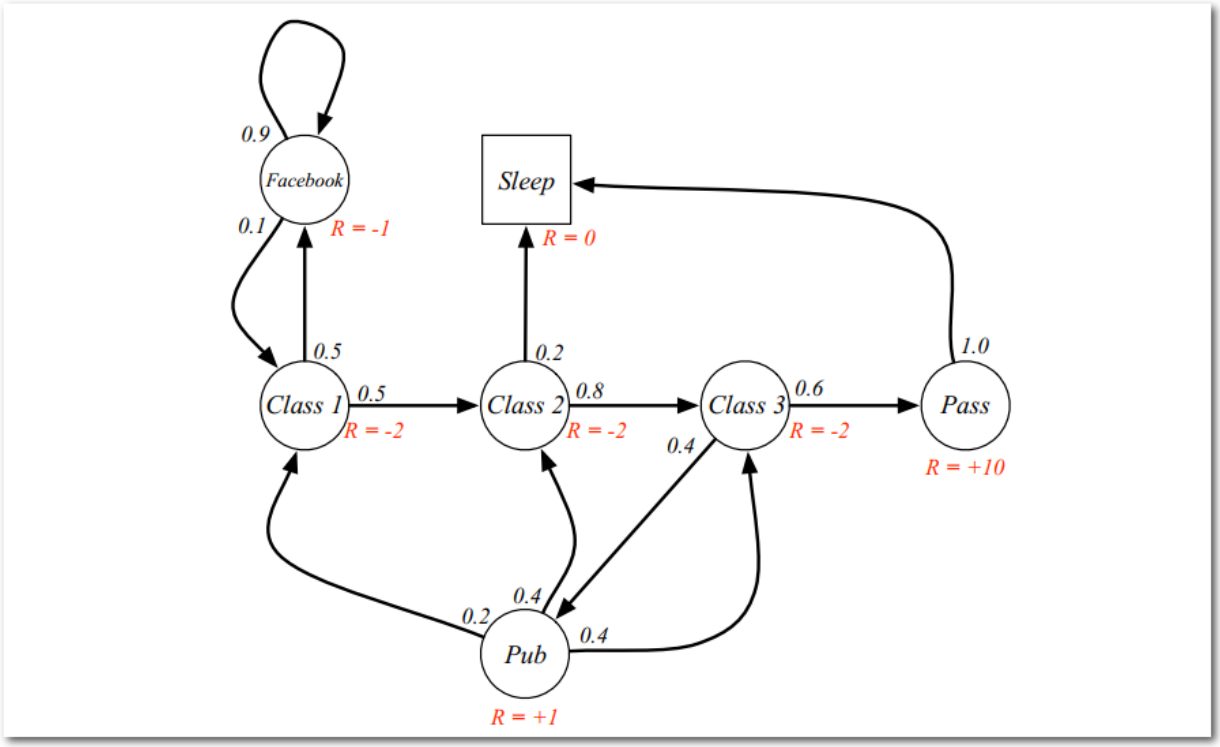
- $\mathcal{S}$ is a finite set of states
- $\mathcal{P}$ is a state transition probability matrix, $\mathcal{P}_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s]$
- $\mathcal{R}$ is a reward function, $\mathcal{R}_s = \mathbb{E}[R_{t+1} \mid S_t = s]$
- γ is a discount factor, $\gamma \in [0,1]$

注：奖励函数$\mathcal{R}$中奖励是t+1时刻的R_{t+1}而不是R_t，这里我们更倾向于理解为：离开这个状态才能获得奖励而不是进入这个状态即获得奖励。David指出这仅是一个约定，为了在描述RL问题中涉及到的观测O、行为A、和奖励R时比较方便。他同时指出如果把奖励改为 R_t 而不是 R_{t+1} ，只要规定好，本质上意义是相同的，在表述上可以把奖励描述为“当进入某个状态会获得相应的奖励”。

第二种解释：对t+1时刻的reward进行索引而不是t时刻，这种方式跟我们如何思考environment和agent之间的界限有关，它的思想是：我们采取action进入环境，然后一个时间步就产生了，我们改

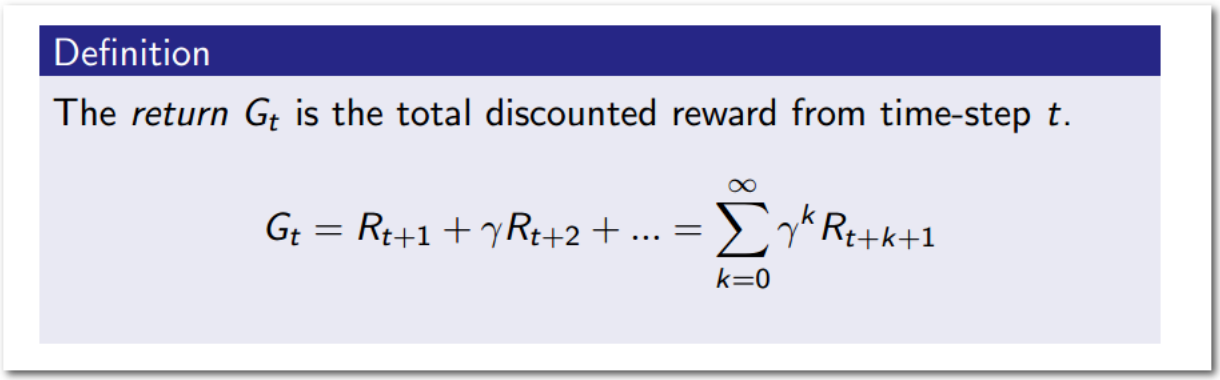
变时间下标是在控制权由环境传递回来之后的新的时间步，我们从环境中的任何东西都会有一个新的下标，包括agent.在环境发生改变之后，任何东西的下标都是t+1

2.1.2 Example: Student MRP



2.2 Return 收获

收获 G_t为在一个马尔科夫奖励链上从t时刻开始往后所有的奖励的有衰减的总和。其中衰减系数体现了未来的奖励在当前时刻的价值比例，在k+1时刻获得的奖励R在t时刻的体现出的价值是 $\gamma^k R$，γ接近0，则表明趋向于“近视”性评估；γ接近1则表明偏重考虑远期的利益。



2.3 Value Function 价值函数

2.3.1 Definition

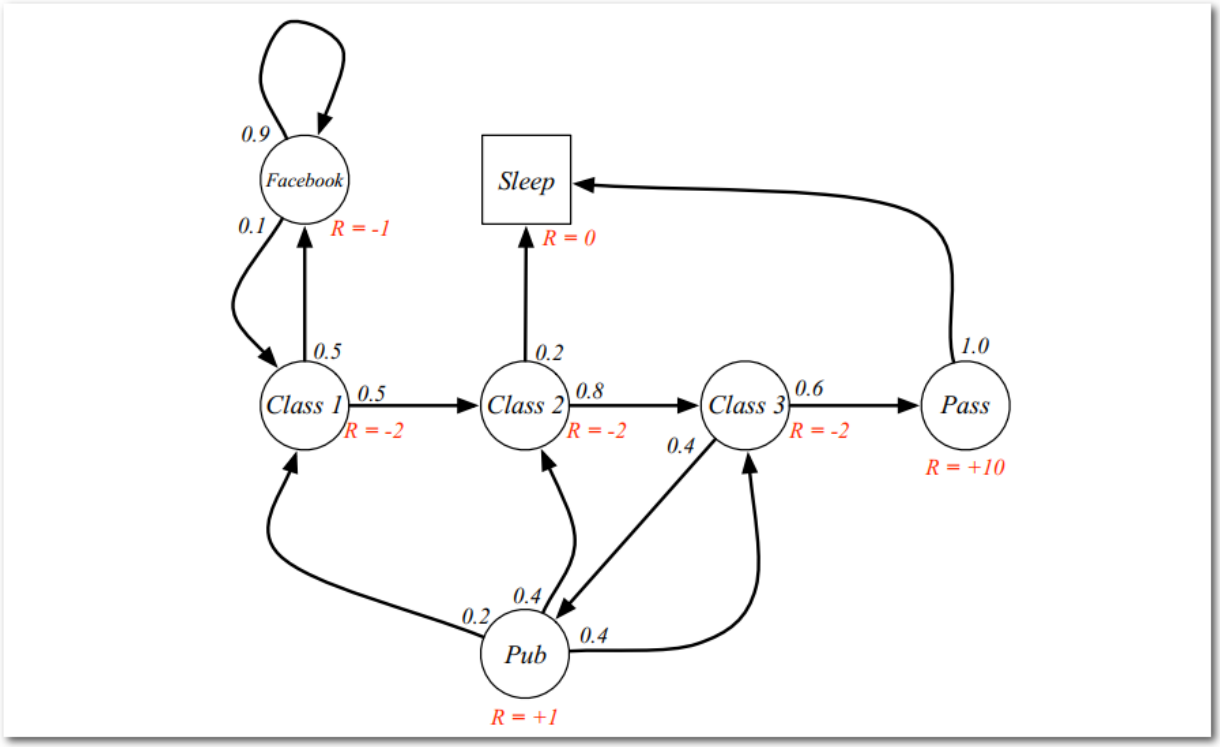
价值函数给出了某一状态或某一行为的长期价值。一个马尔科夫奖励过程中某一状态的价值函数为从该状态开始的马尔可夫链收获的期望。为什么会有期望？因为从t时刻到终止状态的马尔科夫链不止一条，每一条都有对应的概率和Return收益，所以对应的概率乘以相应的收益自然就会有期望。

Definition

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

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

2.3.2 Example: Student MRP Returns



计算收获和价值:

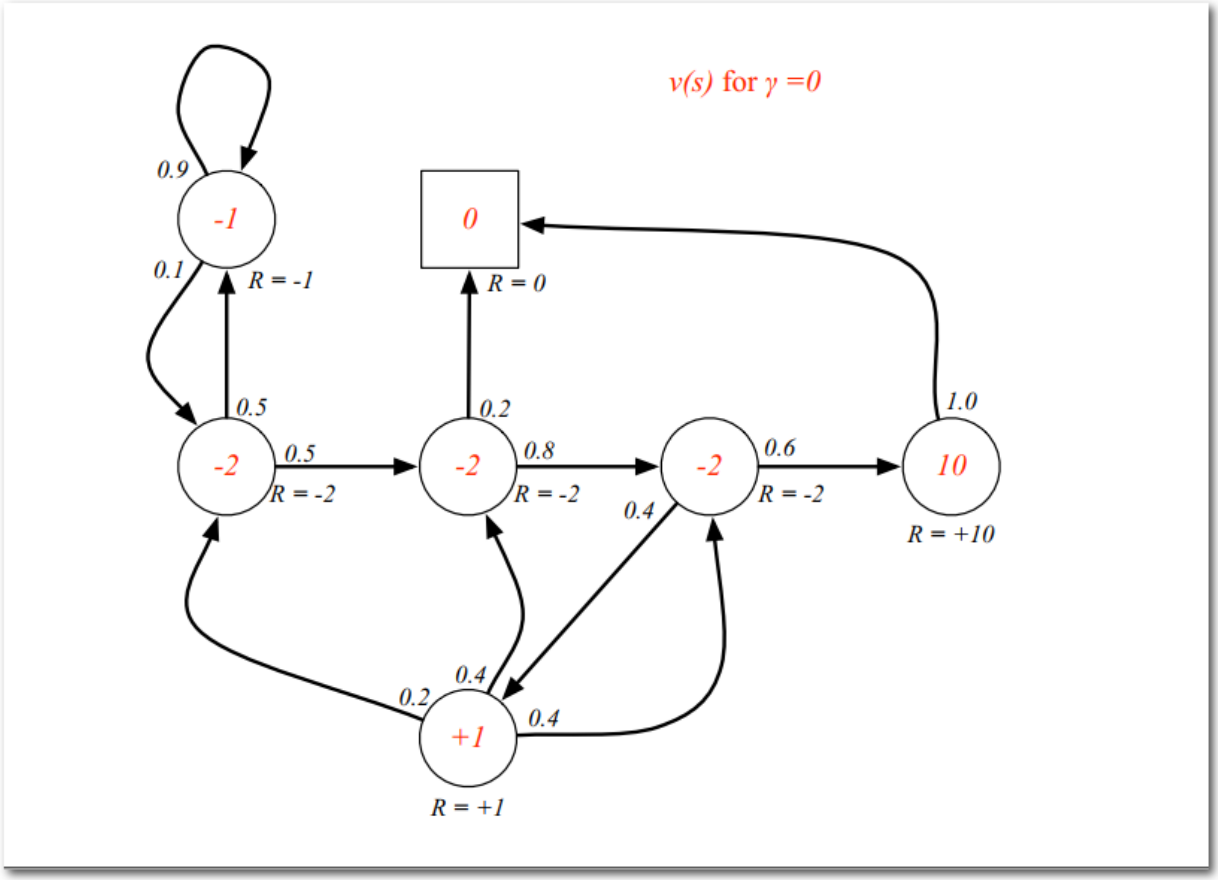
Sample returns for Student MRP:
Starting from $S_1 = \text{C1}$ with $\gamma = \frac{1}{2}$

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

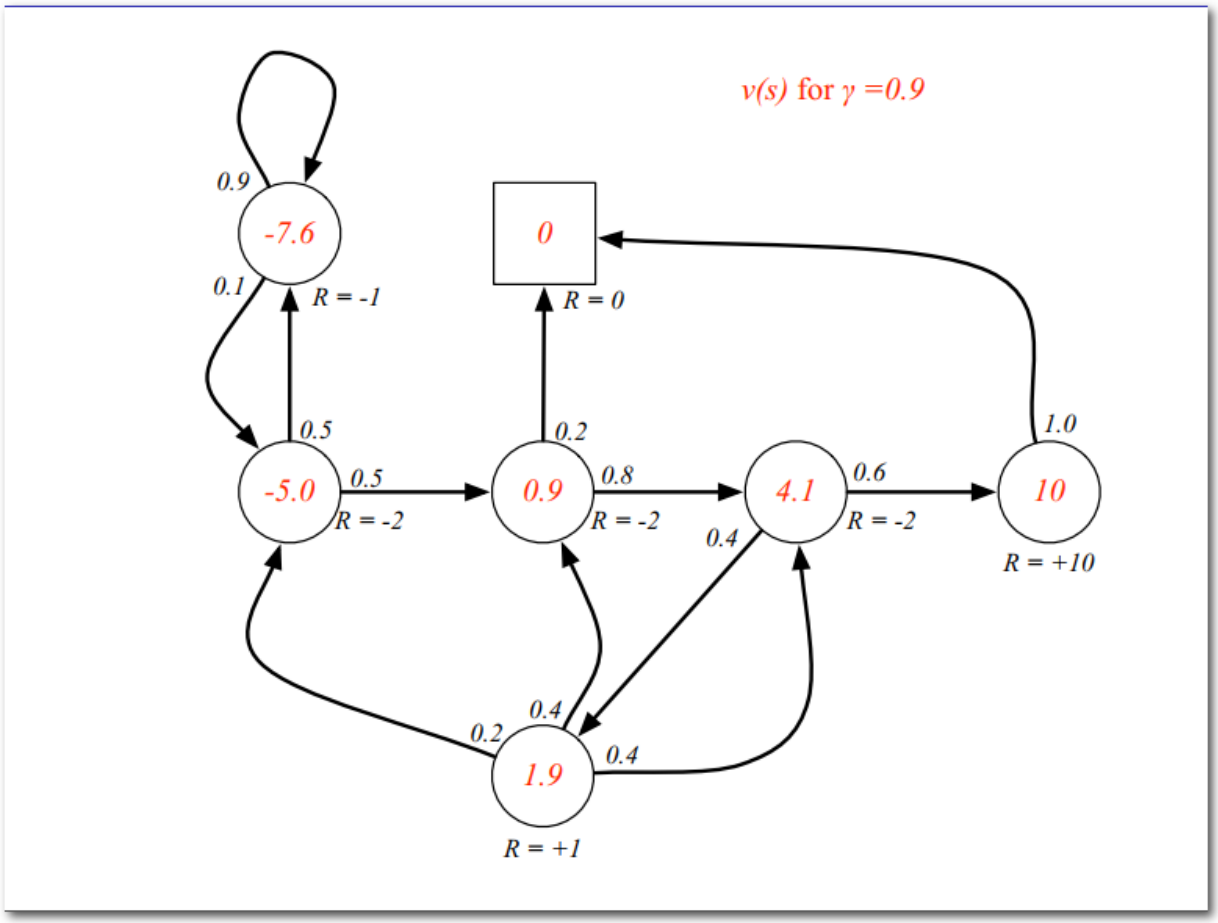
C1 C2 C3 Pass Sleep	$v_1 = -2 - 2*\frac{1}{2} - 2*\frac{1}{4} + 10*\frac{1}{8}$	=	−2.25
C1 FB FB C1 C2 Sleep	$v_1 = -2 - 1*\frac{1}{2} - 1*\frac{1}{4} - 2*\frac{1}{8} - 2*\frac{1}{16}$	=	−3.125
C1 C2 C3 Pub C2 C3 Pass Sleep	$v_1 = -2 - 2*\frac{1}{2} - 2*\frac{1}{4} + 1*\frac{1}{8} - 2*\frac{1}{16}\ldots$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

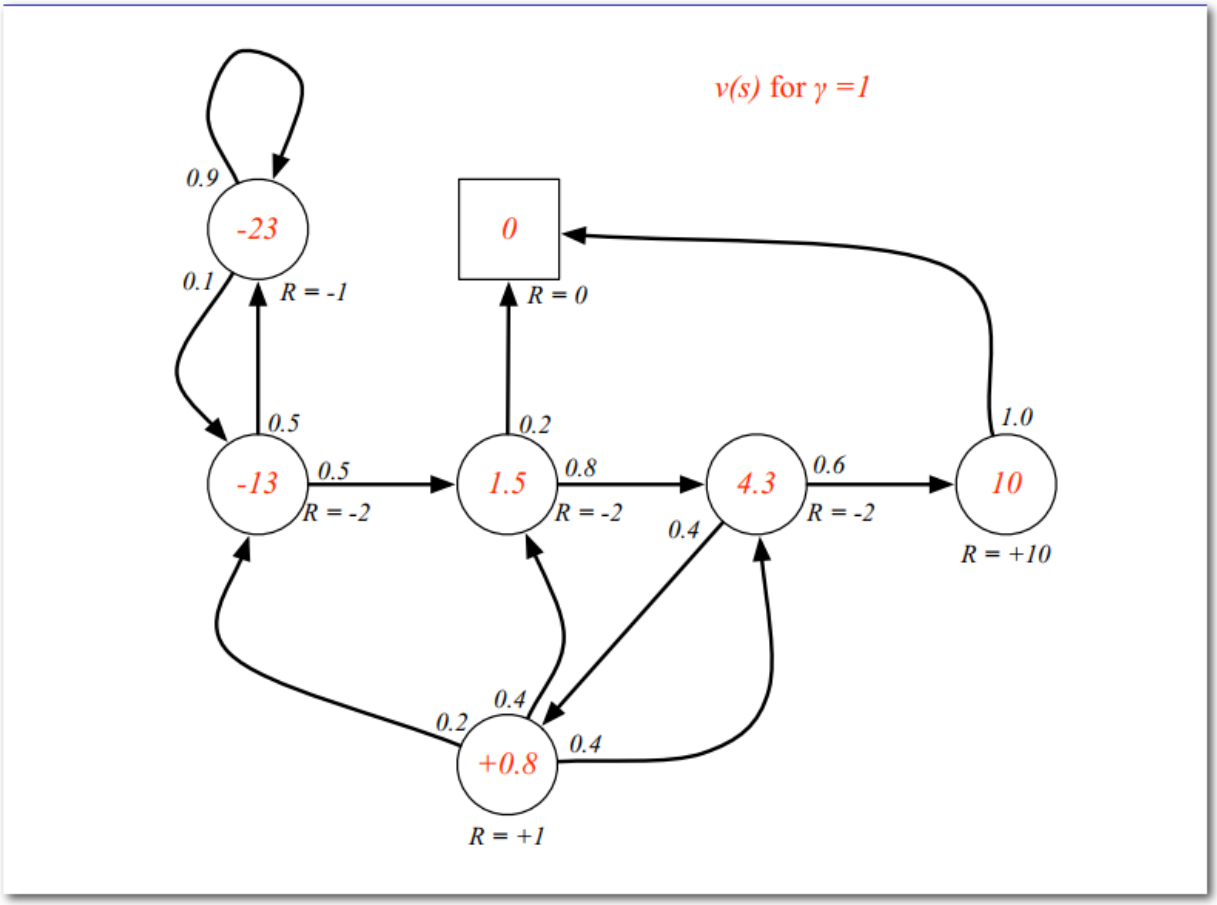
2.3.3 Example: State-Value Function for Student MRP

各状态圈内的数字表示该状态的价值，圈外的R=-2等表示的是该状态的即时奖励。



当γ=0时，是收获的最大短视，各状态的价值与该状态的即时奖励相同





2.4 Bellman Equation 贝尔曼方程

2.4.1 Bellman Equation for MRPs

Bellman方程的基本思想是对value function进行递归分解，主要分为两部分：

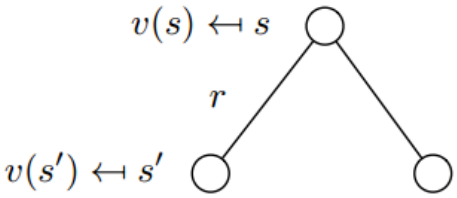
- immediate reward $\mathcal{R}_{t+1}$ (即将得到的)
- discounted value of successor state $\gamma v(S_{t+1})$ (后继state的一个状态到结束状态的value)

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

在导出最后一行时，将 G_{t+1} 变成了 $v(S_{t+1})$，其理由是收获的期望等于收获的期望的期望

backup diagrams:可以看做是向前看一步的探索过程。如果当前我们处于状态s,那么我们可以向前看一步，得到后继状态s'，这样我们可以得到状态的value函数。

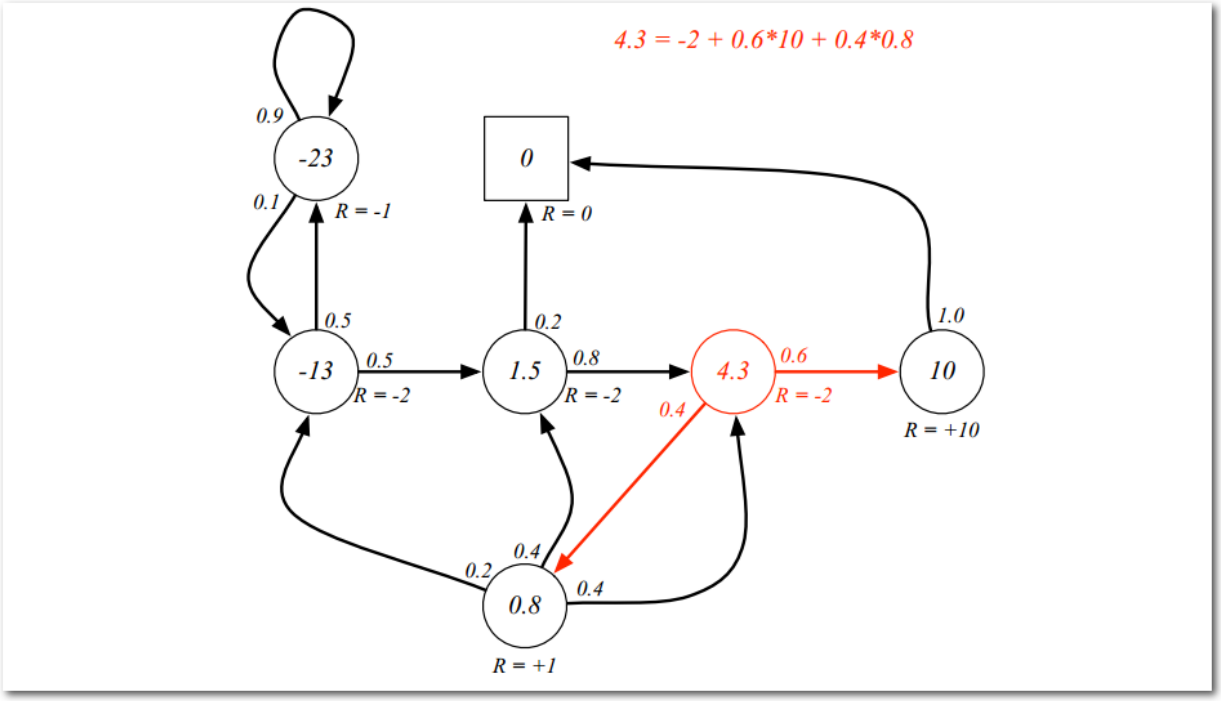
$$v(s) = \mathbb{E}\left[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s\right]$$



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

2.4.2 Bellman Equation for Student MRP

选中一个state，它的value函数的值是4.3。那么，使用Bellman方程来验证这个值。Value 函数应该等价于 向前看了一步，算的是接下来所有可能发生的事得到期望值。从C3出发，有两件事可能发生。首先得到-2分的Reward，这个值是固定的，无论接下来发生什么，都会有得到这个Reward。然后，我们有0.6的概率到达一个value值为10的地方，也有0.4的概率达到pub,pub的value值是0.8,将这些所有的值都加起来，得到-2+0.6*10+0.4*0.8=4.3，结果得到了验证，这确实是MDP的得分。



注：图中γ为1

2.4.3 Bellman Equation in Matrix Form 矩阵形式的Bellman方程

Bellman方程可以用矩阵和向量的形式表示：

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

具体如下：

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

$\mathcal{R}$代表的是immediate reward,$\gamma \mathcal{P} v$代表的是终止状态的value。

2.4.4 Solving the Bellman Equation

Bellman方程是线性方程，所以方程可以被显性的求解：

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

直接求解方程只适用于小规模的MRPs，对于大规模的MRPs这不是一个典型的可操作性强的解决方法。因为如果有n个状态，那么对矩阵求逆的复杂度 是$O(n^3)$,复杂度过高。对于大规模的MRP我们也有许多有效的方法可以提供：

- Dynamic programming 动态规划
- Monte-Carlo evaluation 蒙特卡罗评估
- Temporal-Difference learning 时序差分学习

3. Markov Decision Processes 马尔科夫决策过程

3.1 MDP

3.1.1 Definition 定义

相对于马尔科夫奖励过程，马尔科夫决策过程多了一个行为集合A，它是这样的一个元组: $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$。看起来很类似马尔科夫奖励过程，但这里的$\mathcal{P}$和$\mathcal{R}$都与具体的行为action对应，而不像马尔科夫奖励过程那样仅对应于某个状态，A表示的是有限的行为的集合。具体的定义如下：

Definition

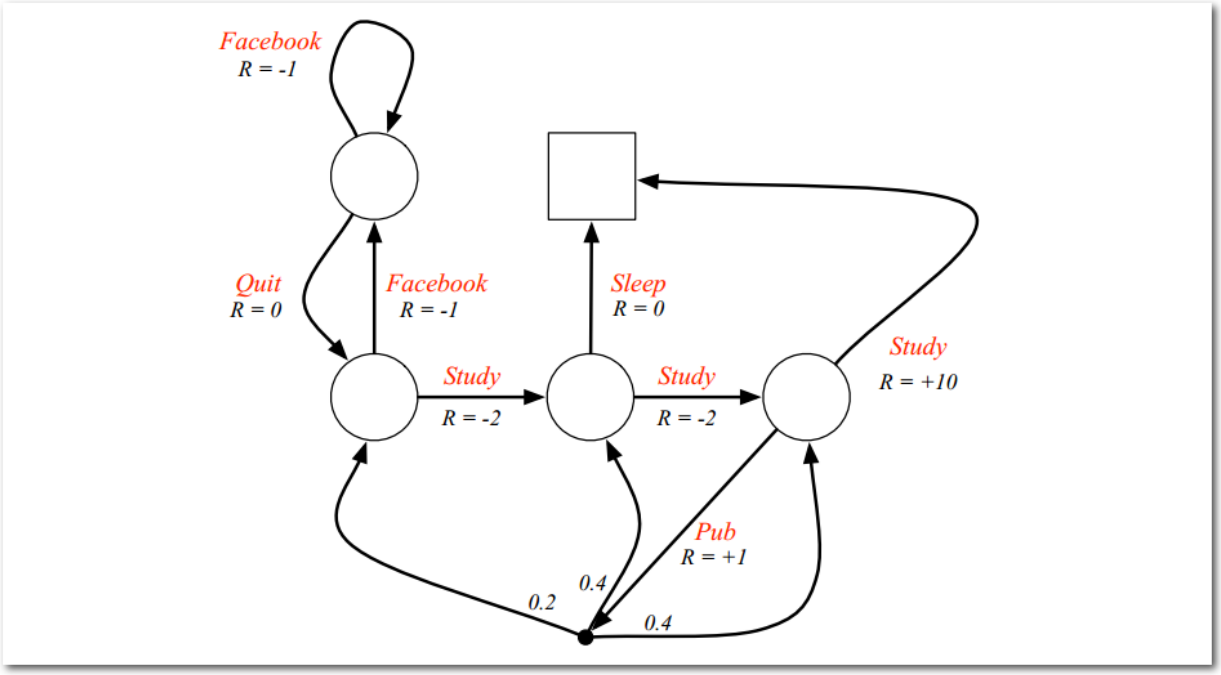
A *Markov Decision Process* is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ 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 $\gamma \in [0, 1]$.

3.1.2 Example: Student MDP

下图给出了一个可能的MDP的状态转化图。图中红色的文字表示的是采取的行为，而不是先前的状态名,图中用黑色小实点表示临时状态，进入临时状态后个体只能被动的被环境按照其动力学分配到另外三个状态，也就是说此时Agent没有选择权决定去哪一个状态。

对比之前的学生MRP示例可以发现，即时奖励与行为对应了，同一个状态下采取不同的行为得到的即时奖励是不一样的。在下图中，如果选择了去酒吧，那么之后出现的state就是随机的了，你可以选择回到第一讲，或者是回到第二讲，或者是第三讲。 现在对于你所要走的路径有了更多的控制权，这个一个agent所面临过的实际情况。现在的目标是在决策的过程中，找到一条最佳的路径，这条路径可以最大化你得到的reward。



3.2 Policies

策略π是在给定state的情况下，一个关于action的概率分布。其元素 $\pi(a|s)$ 为对过程中的某一状态s采取可能的行为a的概率。用 $\pi(a|s)$表示。

Definition

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

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

- 一个策略完整定义了个体的行为方式，也就是说定义了个体在各个状态下的各种可能的行为方式以及其概率的大小。
- policy仅和当前的状态有关，与历史信息无关；现在我们只考虑静态的policies，也就是说，在MDP中，不管我们在哪个时间步(到达这个状态)，我们所采取的措施都是一样的。policy所依赖的唯一一个因素是我们当前所处的state，跟时间步无关：$A_t \sim \pi(\cdot|S_t), \forall t > 0$。
- 在MRP和MDP中，Markov property意味着state s完全特征化了从这个state开始以后的演化过程。

当给定一个MDP:$\mathcal{M} =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ 和一个策略π时，我们遵循一个特定的流程，抽取得到一个特定的states序列 $S_1, S_2, \ldots$，这个序列就是一个是Markov chain,这个过程是一个马尔科夫过程$< S, P^{\pi} >$；同样，状态和奖励序列$S_1, R_2, S_2, R_3, S_3, \ldots$是一个马尔科夫奖励过程 $< S, P^{\pi}, R^{\pi}, \gamma >$，并且在这个奖励过程中满足下面两个方程：

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

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

在执行策略π时，状态从s转移至 s' 的概率等于一系列概率的和，这一系列概率指的是在执行当前策略时，执行某一个行为的概率与该行为能使状态从s转移至s'的概率的乘积。

当前状态s下执行某一指定策略得到的即时奖励是该策略下所有可能行为得到的奖励与该行为发生的概率的乘积的和。

策略在MDP中的作用相当于agent可以在某一个状态时做出选择，进而有形成各种马尔科夫过程的可能，而且基于策略产生的每一个马尔科夫过程是一个马尔科夫奖励过程，各过程之间的差别是不同的选择产生了不同的后续状态以及对应的不同的奖励。

3.3 Value Functions 基于策略π的价值函数

3.3.1 state-value function $v_{\pi}(s)$ 状态价值函数

$v_{\pi}(s)$ 是在MDP下的基于策略π的**状态价值函数**，表示从状态s开始，遵循当前策略时所获得的收获的期望；或者说在执行当前策略π时，衡量个体处在状态s时的价值大小。具体定义表示如下：

Definition

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

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

注意策略是静态的、关于整体的概念，不随状态改变而改变；变化的是在某一个状态时，依据策略可能产生的具体行为，因为具体的行为是有一定的概率的，策略就是用来描述各个不同状态下执行各个不同行为的概率。

3.3.2 action-value function$q_\pi(s,a)$ 行为价值函数

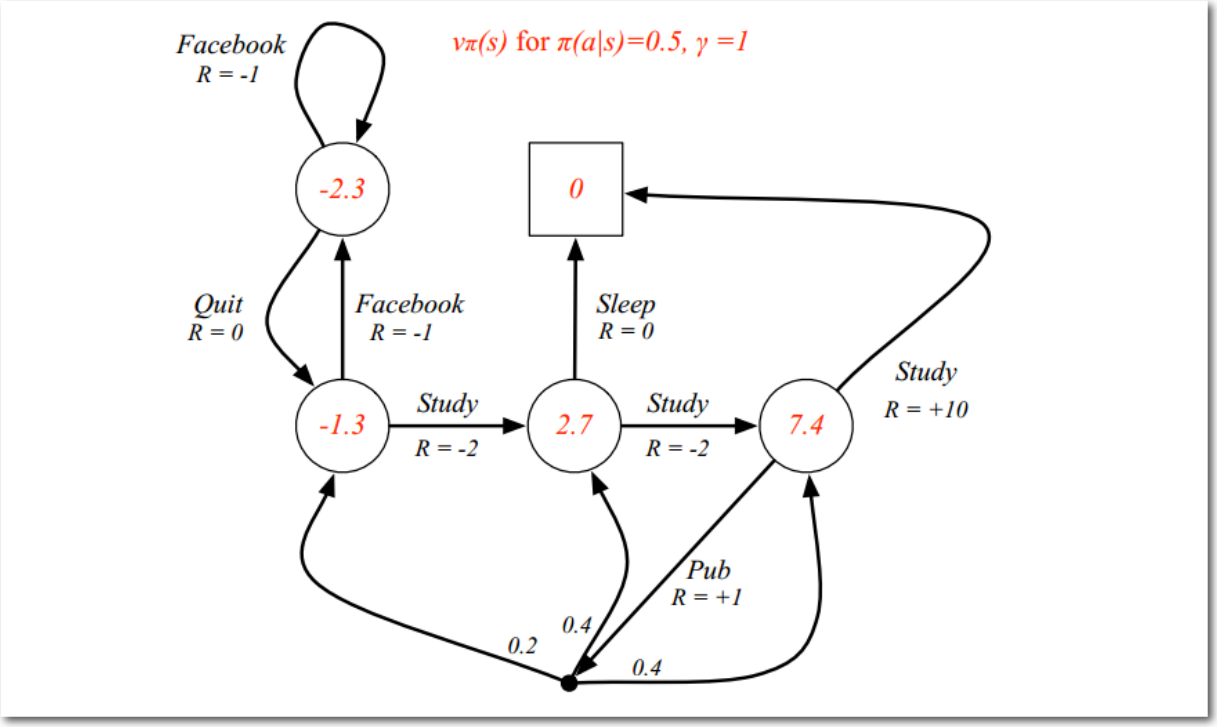
$q_\pi(s,a)$为**行为价值**函数，表示在执行策略π时，对当前状态s执行某一具体行为a所能得到的收获的期望；或者说在遵循当前策略π时，衡量对当前状态执行行为a的价值大小。行为价值函数一般都是与某一特定的状态相对应的，更精细的描述是**状态行为对**价值函数。行为价值函数的公式描述如下：

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

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

3.3.3 Example: State-Value Function for Student MDP



3.4 Bellman Expectation Equation

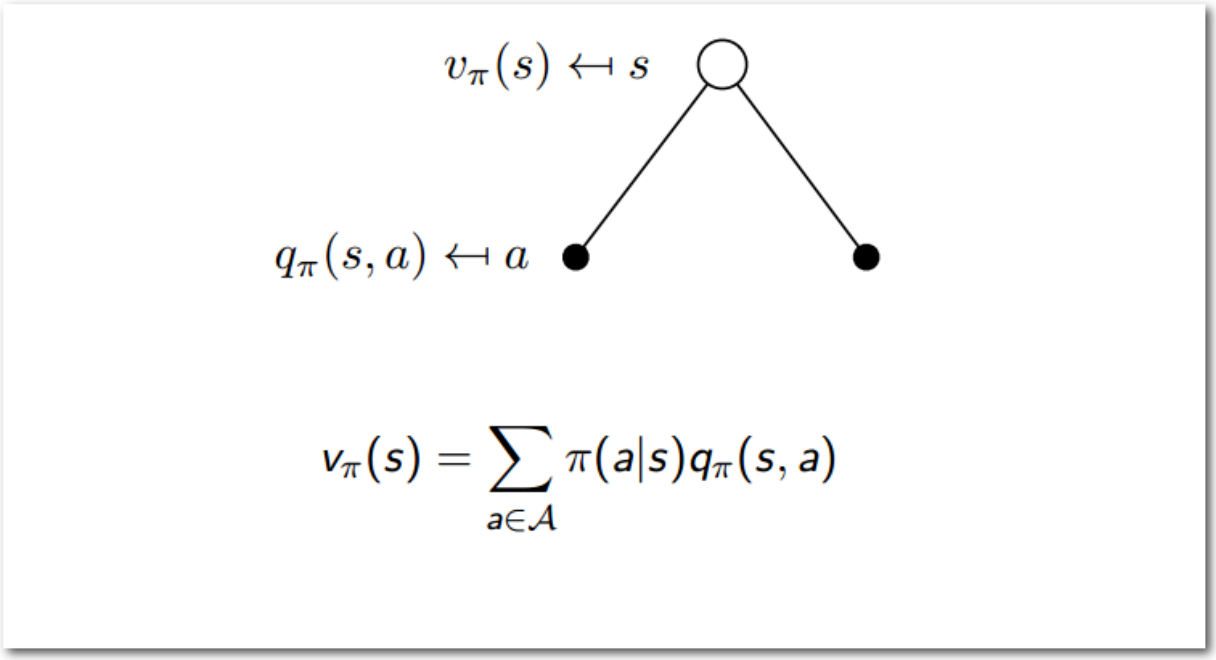
MDP下的状态价值函数和行为价值函数与MRP下的价值函数类似，我们可以使用同样的方法来分解value-function.state-value function可以用如下公式表达：

$$v_\pi(s) = E_\pi[R_{t+1} + \gamma v_\pi(S_t + 1)|S_t = s)]$$

action-value function可用如下公式表达：

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

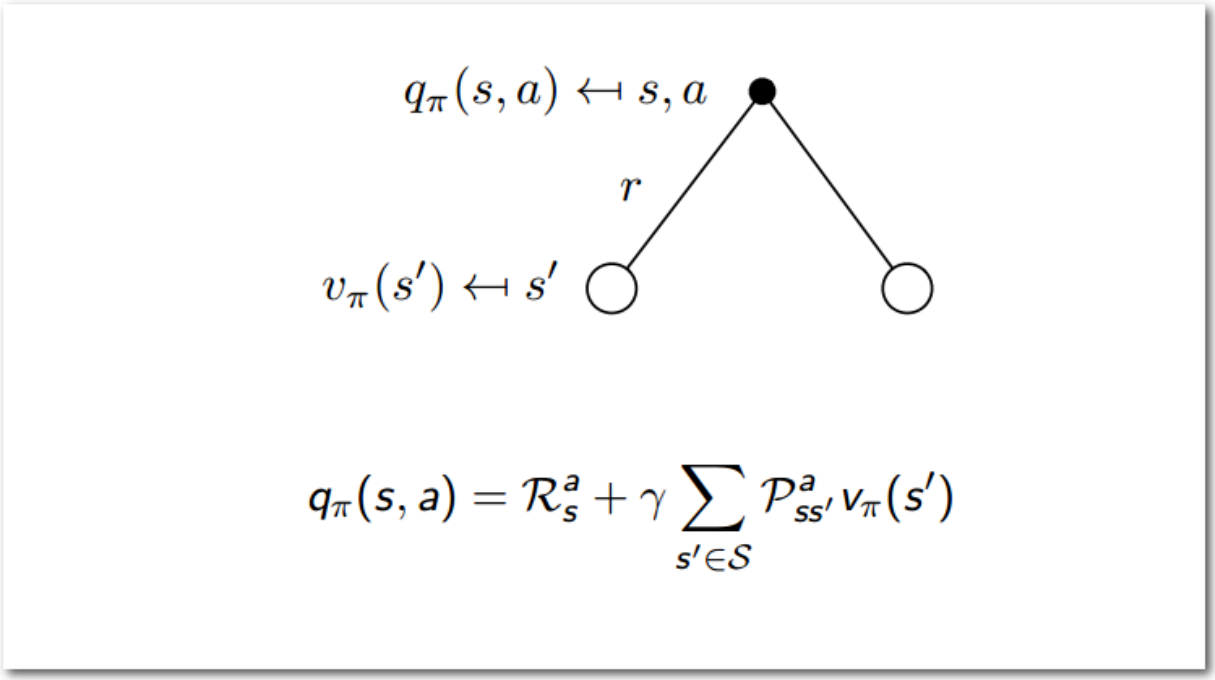
3.4.1 Bellman Expectation Equation for V π



注：空心圆圈表示状态，黑色实心小圆表示的是动作本身，连接状态和动作的线条仅仅把该状态以及该状态下可以采取的行为关联起来。

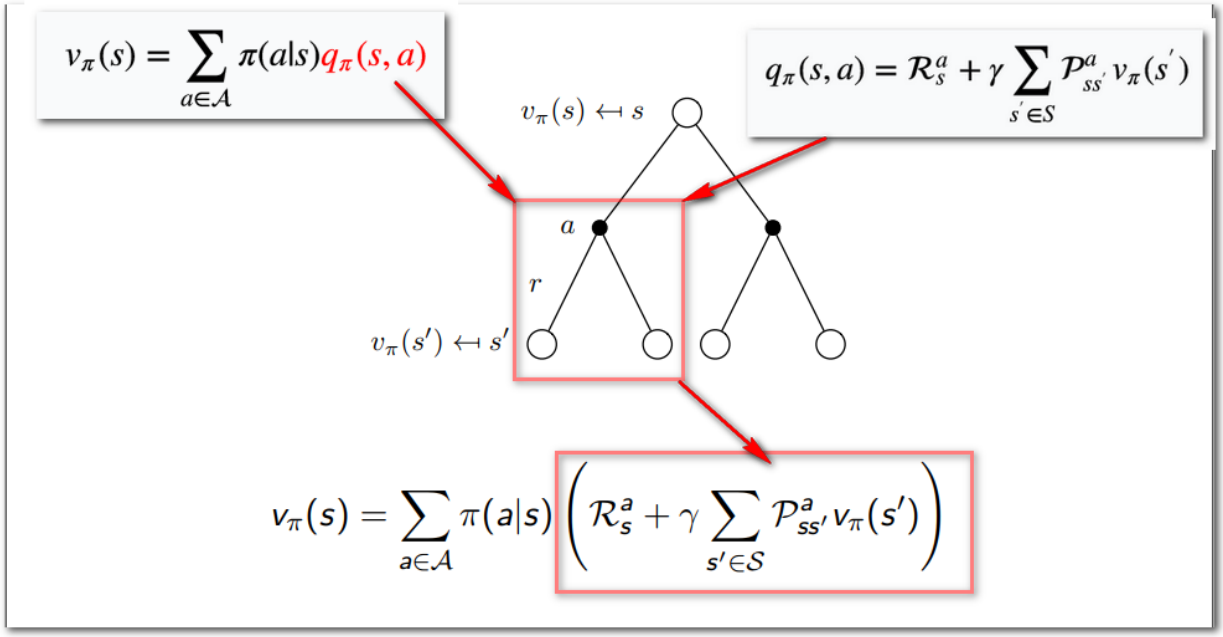
在遵循策略π时，状态s的价值体现为在该状态下遵循某一策略而采取所有可能行为的价值按行为发生概率的乘积求和。

3.4.2 Bellman Expectation Equation for Qπ

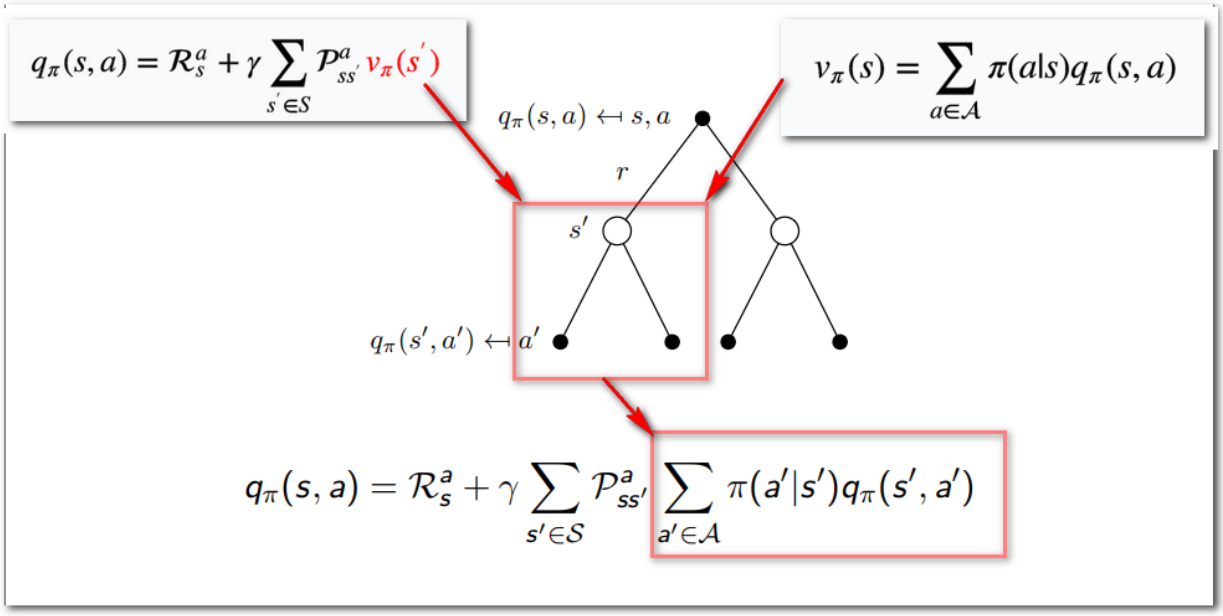


上图表明，某一个状态下采取一个行为的价值，可以分为两部分：其一是离开这个状态的价值，其二是所有进入新的状态的价值于其转移概率乘积的和。***V***告诉我们当它处于一个特定状态时这个state有多好，q告诉我们采取一个特定的action是有多好。

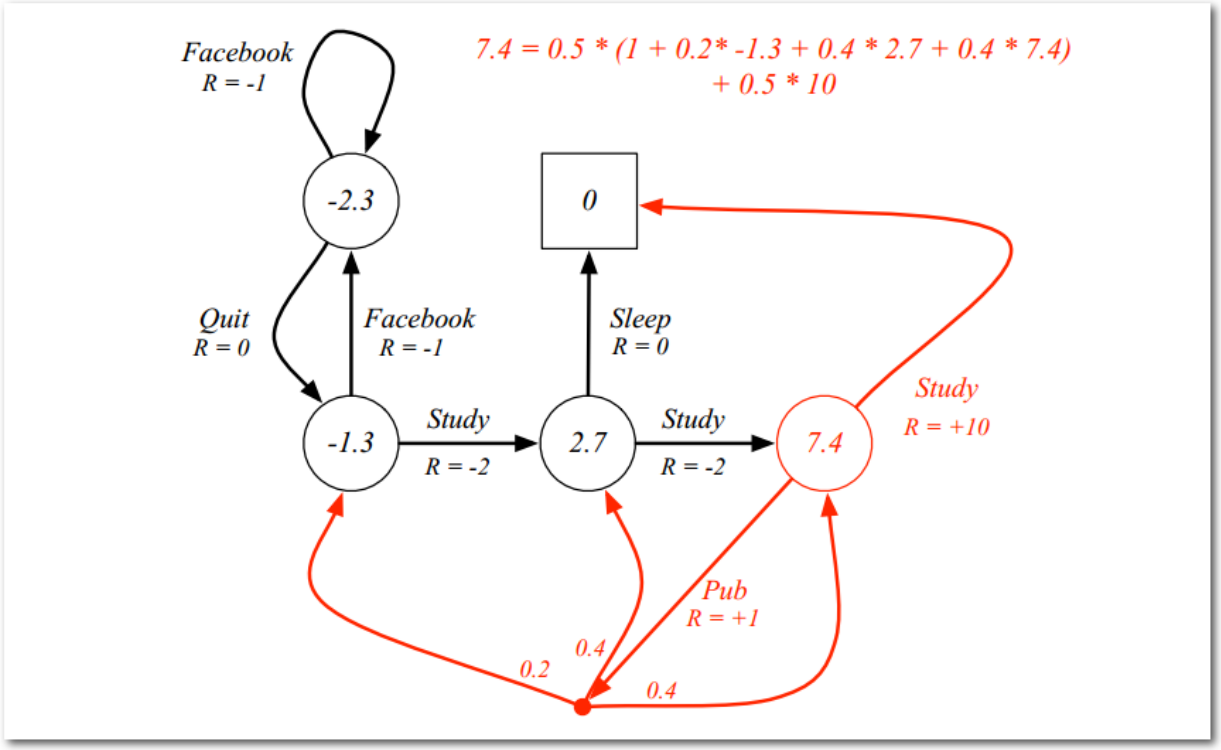
3.4.3 Bellman Expectation Equation for V π (2)



3.4.4 Bellman Expectation Equation for Qπ(2)



3.4.5 Example: Bellman Expectation Equation in Student MDP



上图中的7.4由下面的公式得到：

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

注：我们做一件事的概率是50%,γ为1

3.4.6 Bellman Expectation Equation (Matrix Form) Bellman期望方程矩阵形式

Bellman期望方程矩阵形式：

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

3.5 Optimal Value Function 最优价值函数

3.5.1 Definition

最优状态价值函数 $v_*(s)$指的是从所有策略产生的状态价值函数中，选取使状态s价值最大的函数：

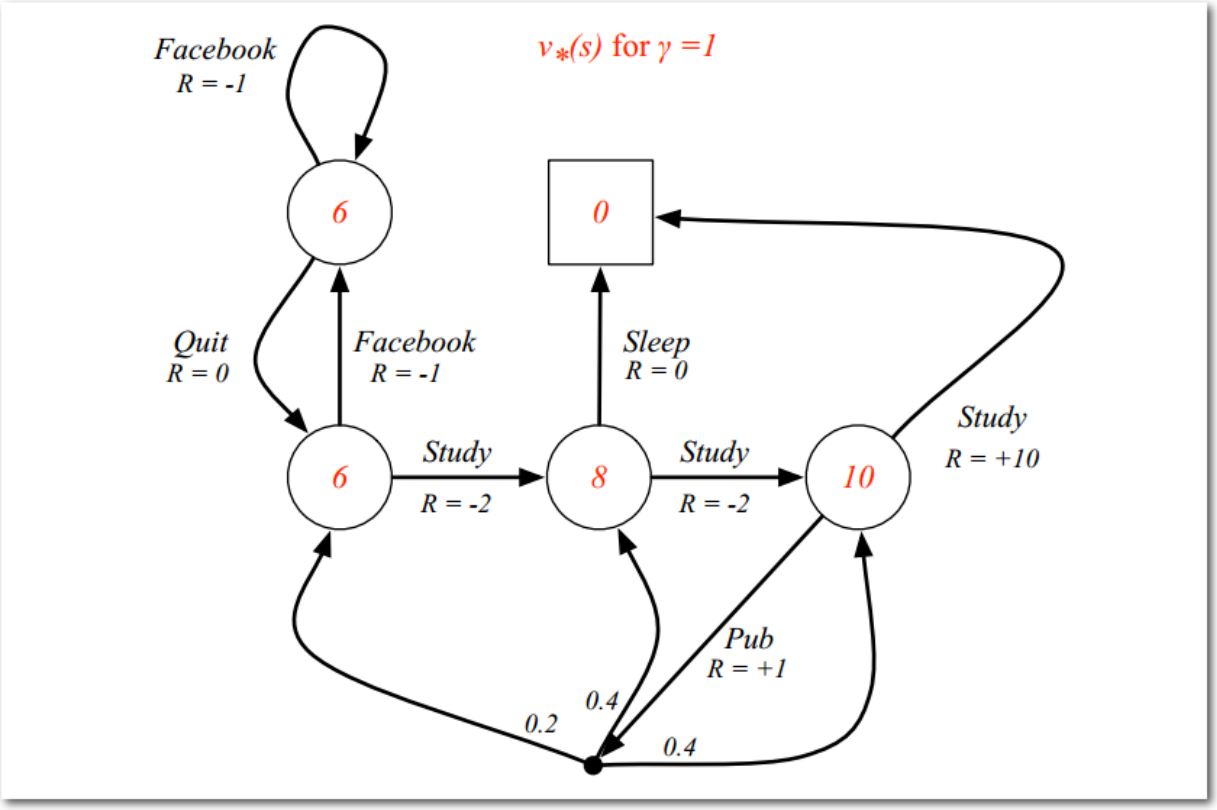
$$v_* = \max_\pi v_\pi(s)$$

类似的，**最优行为价值函数**$q_*(s,a)$ 指的是从所有策略产生的行为价值函数中，选取状态行为对 $< s, a >$ 价值最大的函数：

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

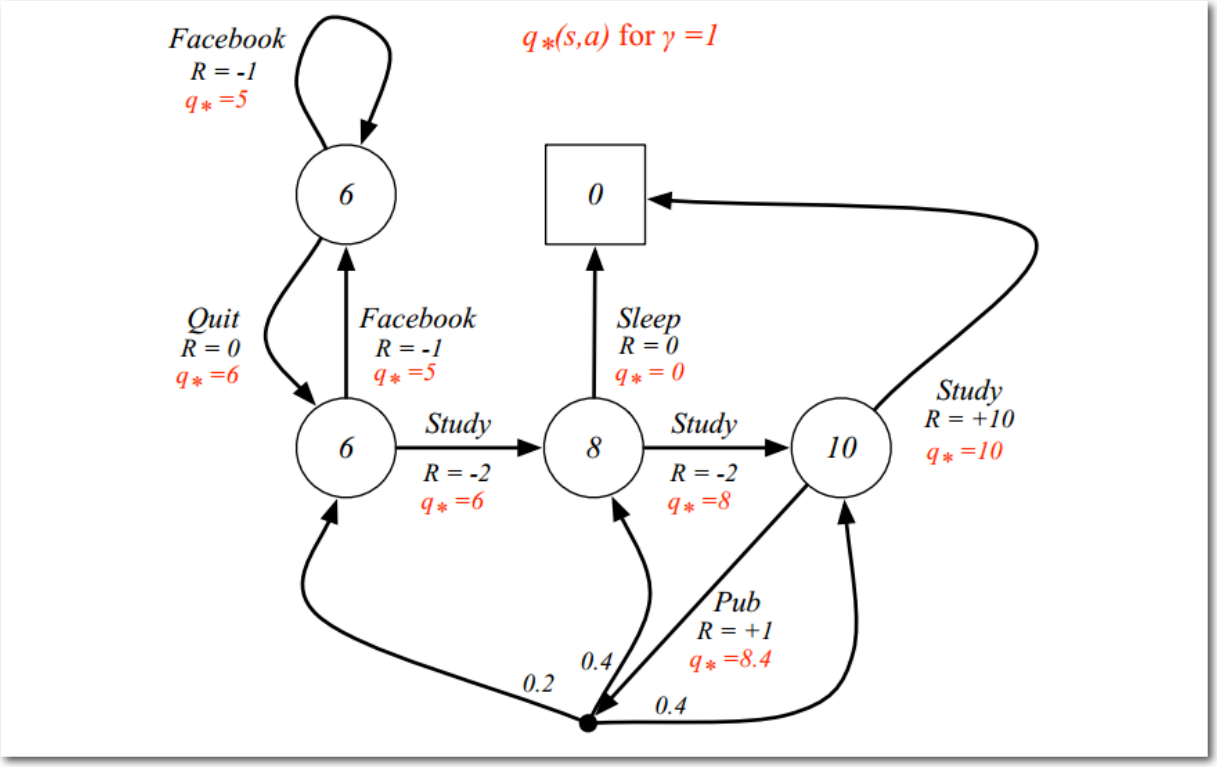
最优价值函数明确了MDP的最优可能表现，当我们知道了最优价值函数，也就知道了每个状态的最优价值，这时便认为这个MDP获得了解决。

3.5.2 Example: Optimal State-Value Function for Student MDP 学生MDP问题的最优状态价值



注：红色的数字代表了每个状态最优的value function

3.5.3 Example: Optimal Action-Value Function for Student MDP 学生MDP问题的最优行为价值



注：Pub行为对应的价值是+9.4而不是+8.4：1+0.4*10+0.4*8+0.2*6=9.4

3.5.4 Optimal Policy 最优策略

对于任何状态s，遵循策略π的价值不小于遵循策略 π' 下的价值，则策略π优于策略 π'：

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

对于任何MDP，下面几点成立：

- 存在一个最优策略π_*，比任何其他策略更好或至少相等:$\pi_* \geq \pi, \forall \pi$
- 所有的最优策略有相同的最优价值函数:$v_{\pi_*}(s) = v_*(s)$
- 所有的最优策略具有相同的行为价值函数:$q_{\pi_*}(s,a) = q_*(s,a)$

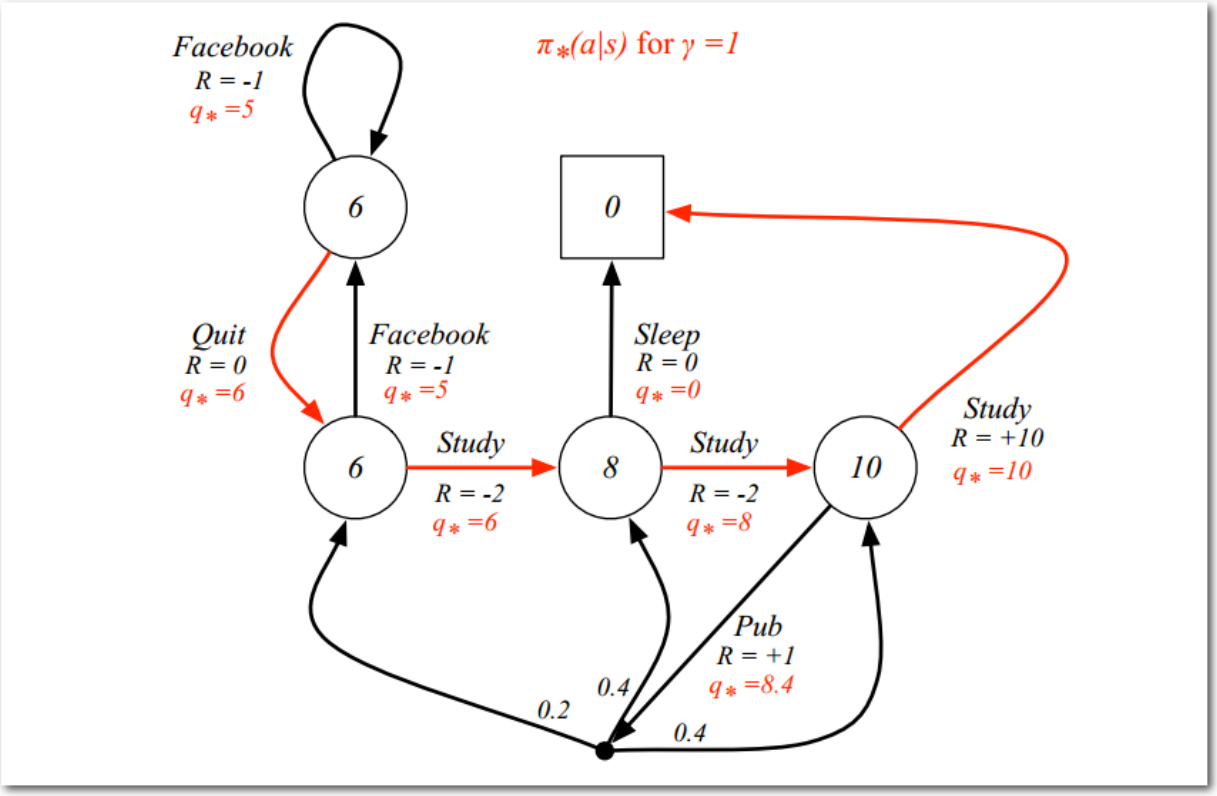
3.5.5 Finding an Optimal Policy 寻找最优策略

寻找最优策略可以从最大化最优行为价值函数来找到：

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

对于任何MDP问题，总存在一个确定性的最优策略；同时如果我们知道最优行为价值函数，则表明我们找到了最优策略。

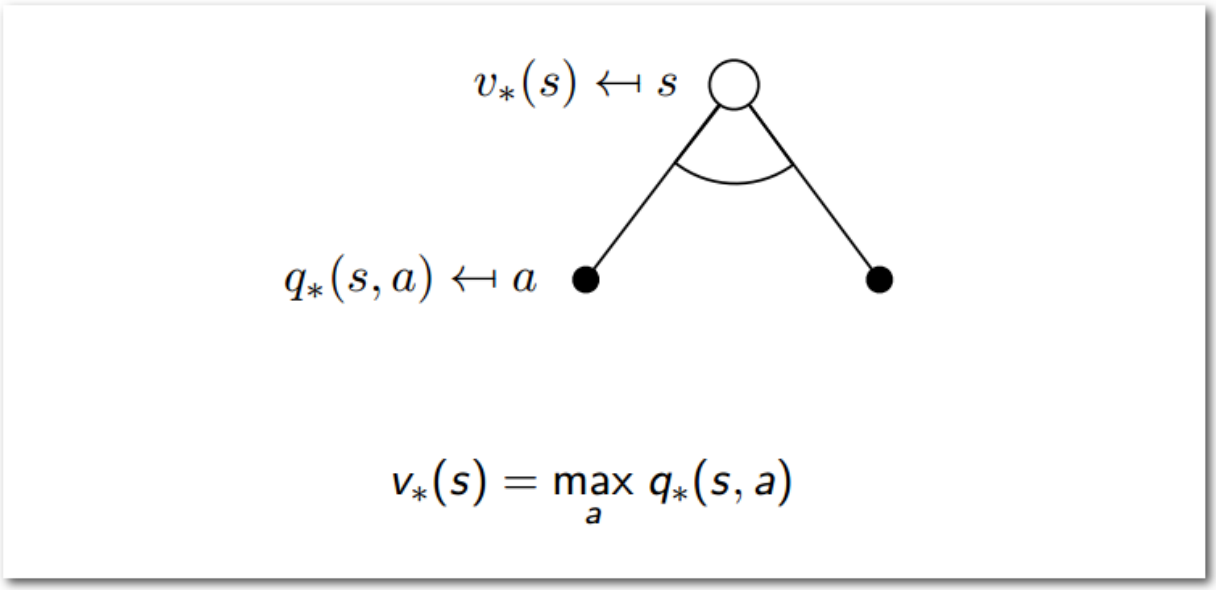
3.5.6 Example: Optimal Policy for Student MDP 学生MDP最优策略示例



注：红色的弧代表在该状态下最优的action value，所有的状态和相应的动作组合起来也就是最优的策略

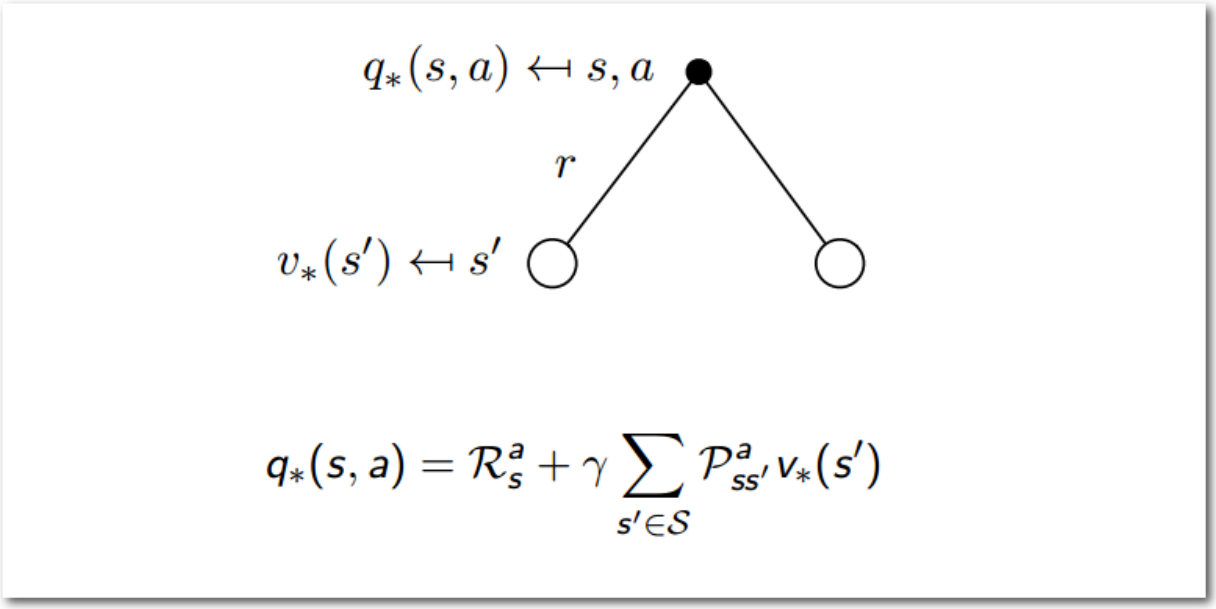
3.5.7 Bellman Optimality Equation for V*

针对v_*，一个状态的最优价值等于从该状态出发采取的所有行为产生的行为价值中最大的那个行为价值：



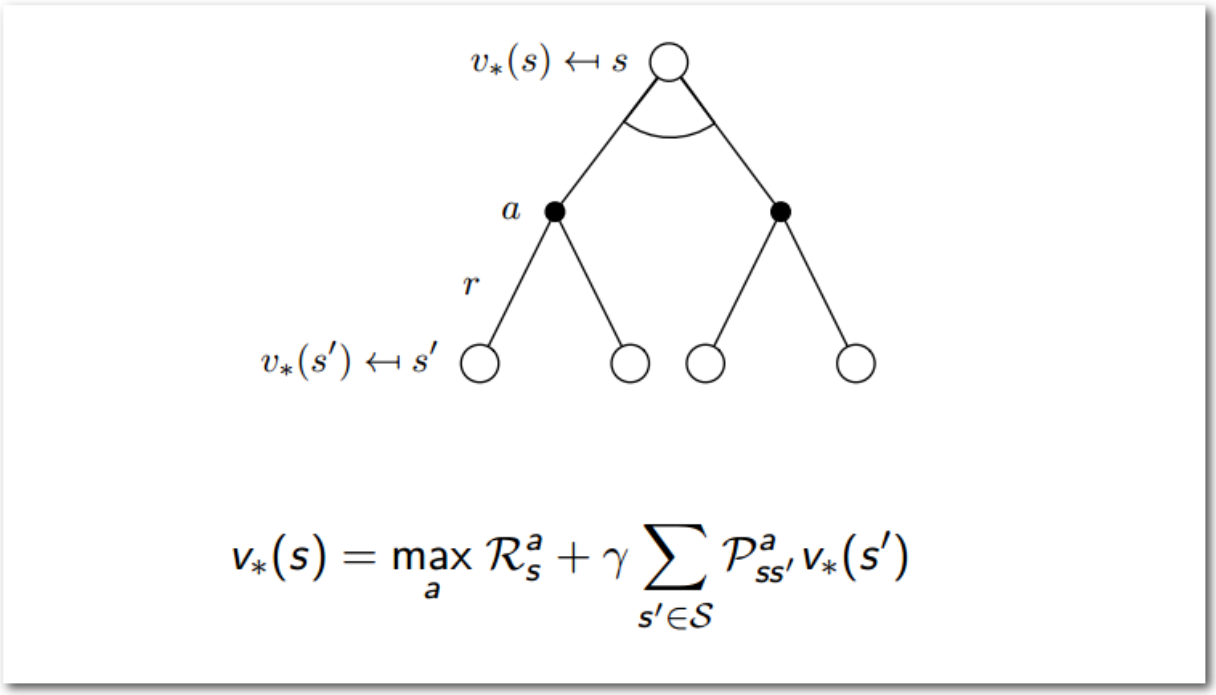
3.5.8 Bellman Optimality Equation for Q*

针对 q_* ，在某个状态s下，采取某个行为的最优价值由2部分组成，一部分是离开状态 s 的即时奖励，另一部分则是所有能到达的状态 s' 的最优状态价值按出现概率求和：



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

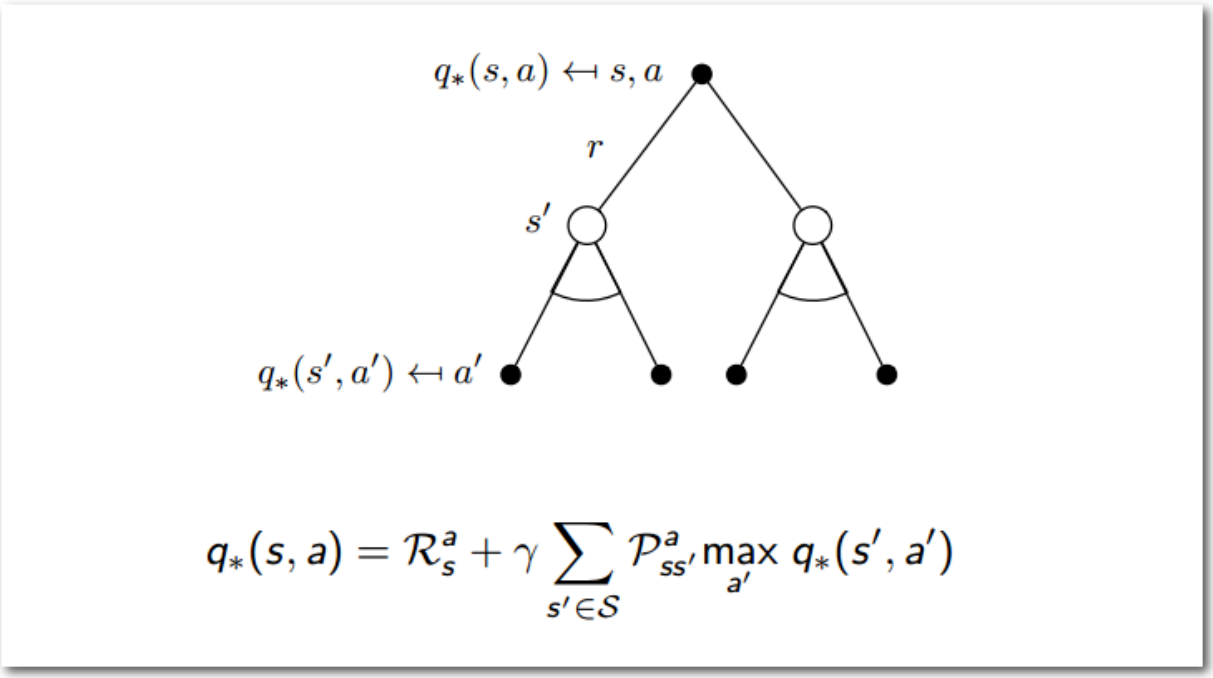
3.5.9 Bellman Optimality Equation for V∗



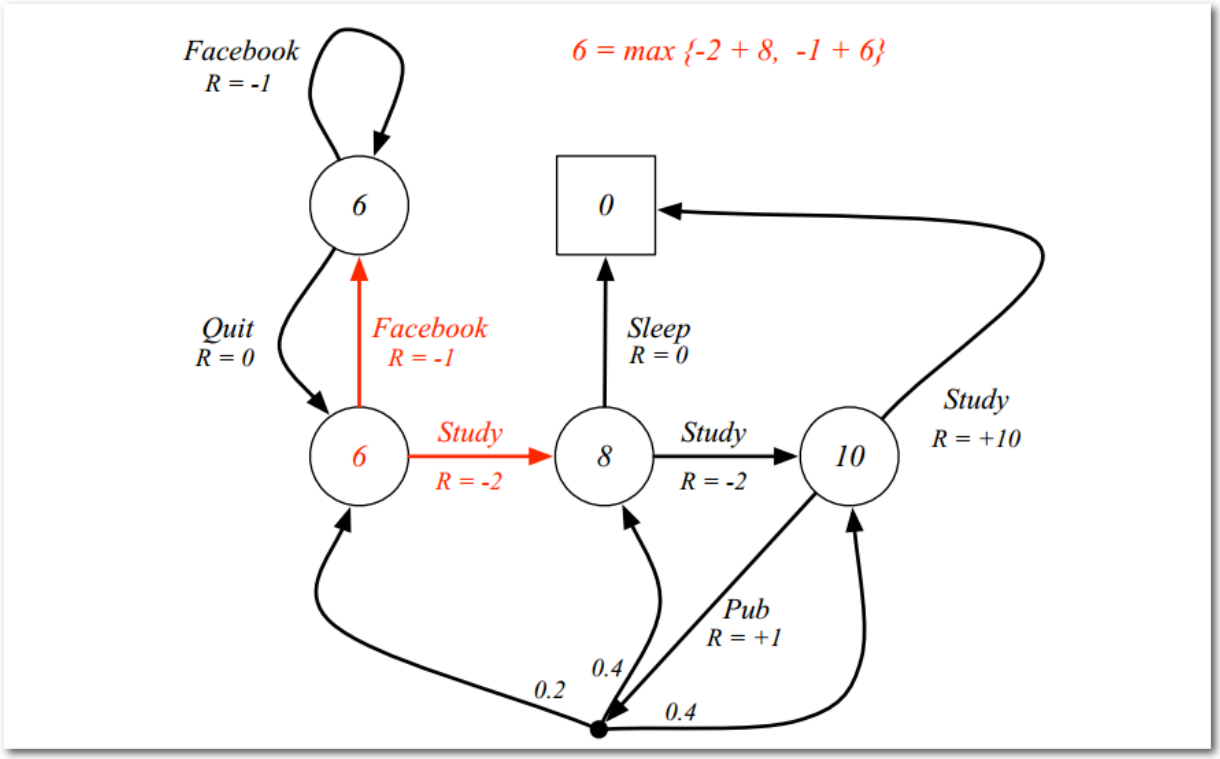
$$v_*(s) = \max_a \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

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

3.5.10 Bellman Optimality Equation for Q∗



3.5.11 Example: Bellman Optimality Equation in Student MDP Bellman最优方程学生MDP示例



3.5.12 Solving the Bellman Optimality Equation 求解Bellman最优方程

Bellman最优方程是非线性的，这个方程的解法不存在一个显式的公式一般采用迭代方法解决该问题，以下是迭代方法最有名的例子：

- Value Iteration
- Policy Iteration
- Q-learning
- Sarsa