强化学习基本原理及编程实现：有限马尔科夫决策过程

郭宪

2019.09.22

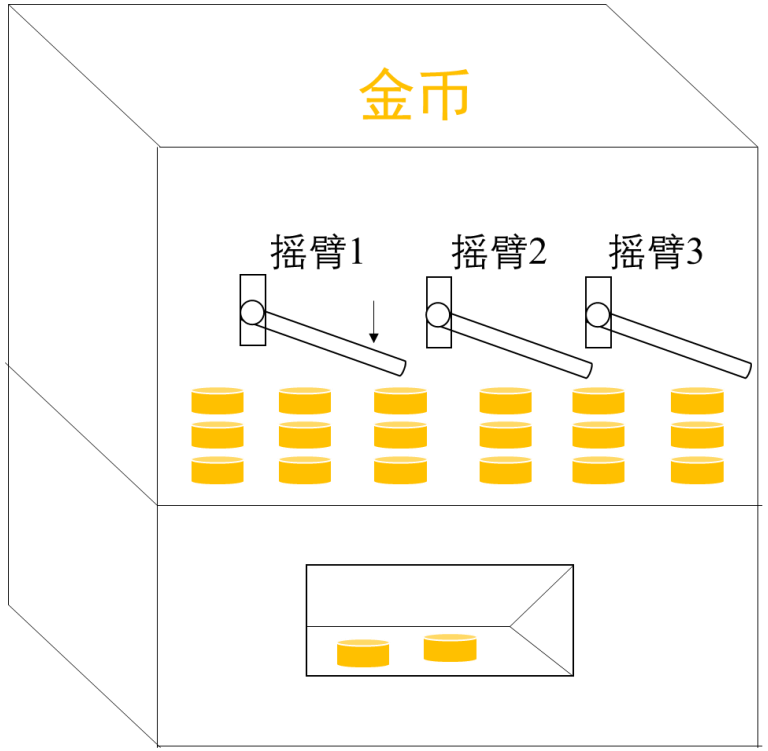
人工智能学院

College of Artificial Intelligence

南開大學

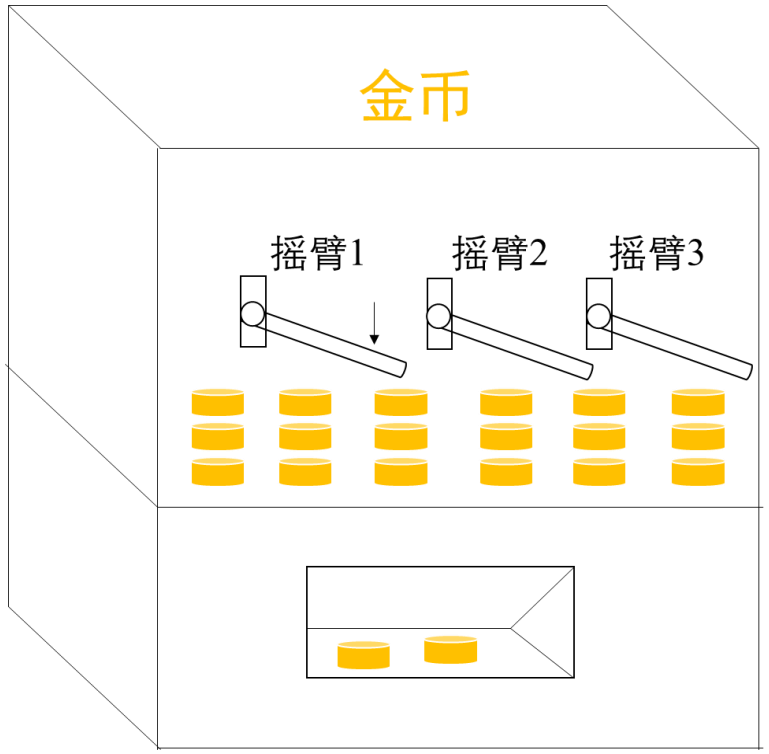
Nankai University

从多臂赌博机到马尔科夫决策过程

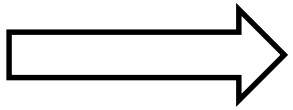


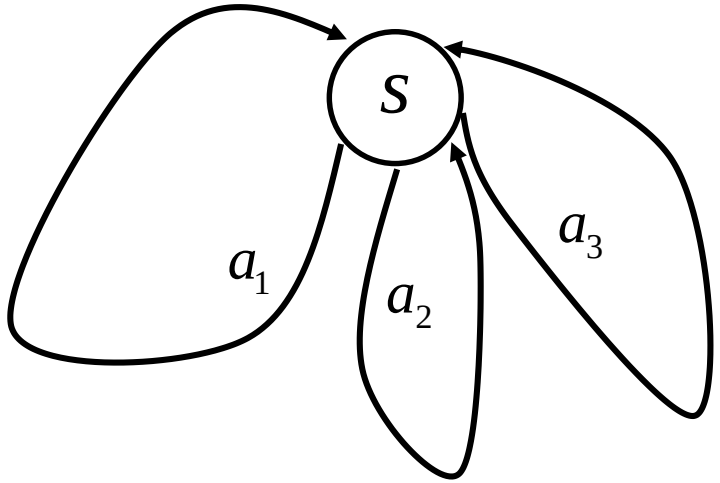
单状态 ⟹ 多状态 ⟹ 状态与状态之间符合马尔科夫性

多臂赌博机的描述



单状态

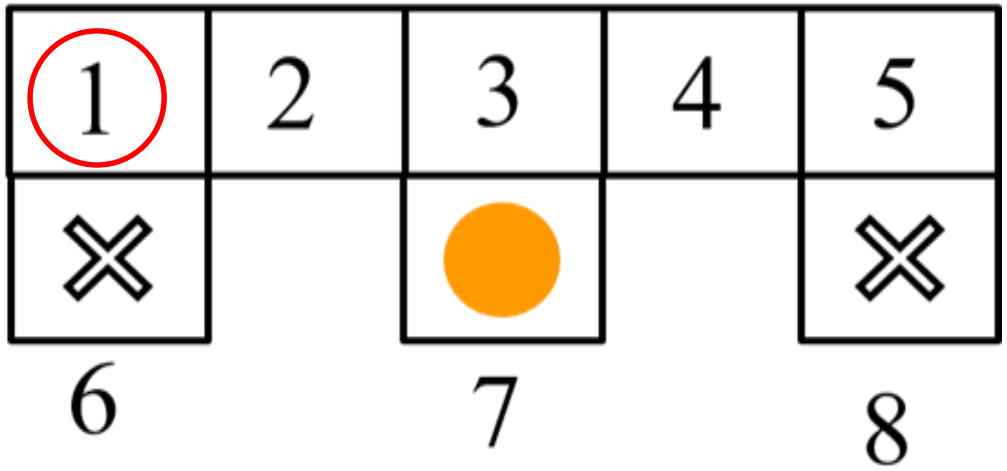




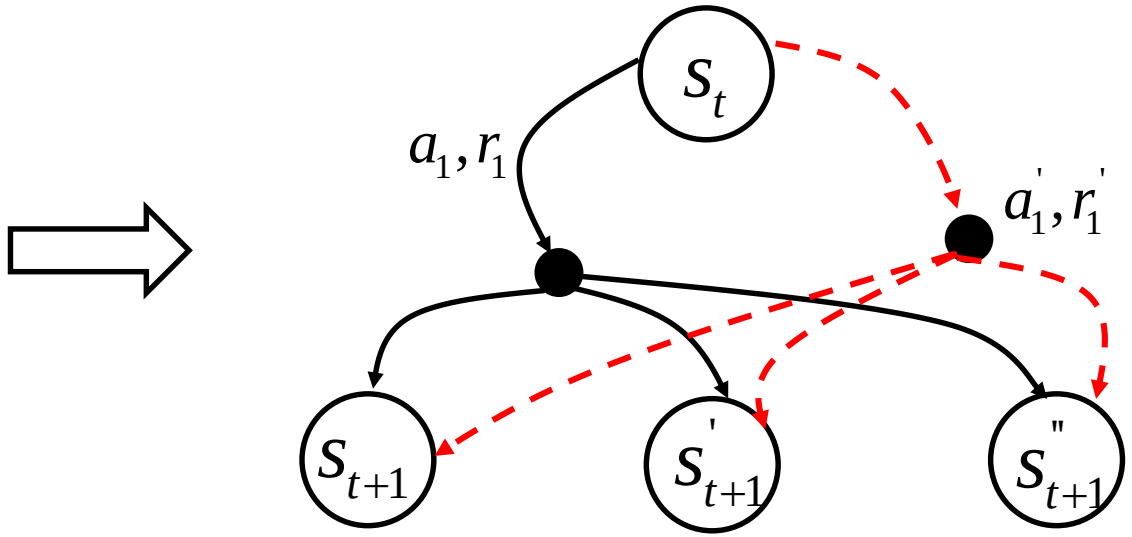
$$s \xrightarrow{a} r, s$$

多臂赌博机

马尔科夫决策过程的直观理解



多状态



$$s_t \xrightarrow{a_t} r_t, s_{t+1}$$

马尔科夫决策过程

动作不仅仅影响回报还影响状态的转移

马尔科夫决策过程描述交互过程

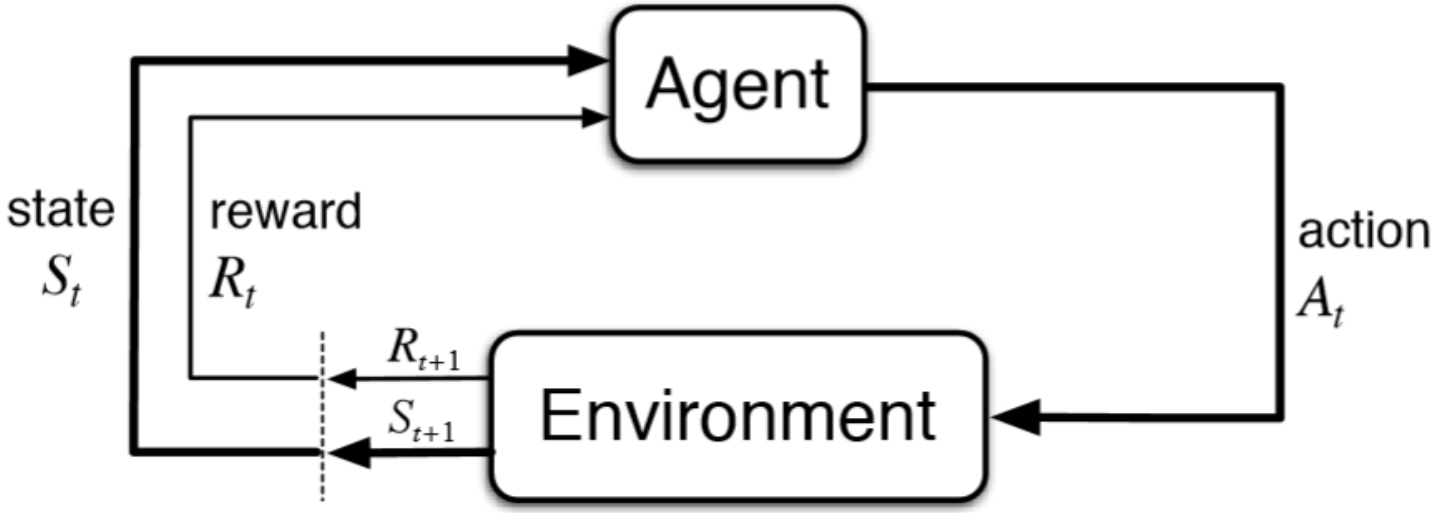
Agent: 智能体

Environment: 环境

智能体与环境进行序贯交互：

trajectory(轨迹)

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \cdots$$



智能体与环境交互

如果将状态 S_t 和回报 R_t 视为随机变量，那么该轨迹可以用一个随机过程来描述

随机变量，条件概率，分布，期望，方差，随机过程，马尔科夫过程的概念

概率论基础

1. 随机变量：

   随机变量是指可以随机地取不同值的变量

2. 概率分布：

   概率分布用来描述随机变量在每个可能取到的值处的可能性大小

3. 期望和方差：

离散型：$$E_{x\sim P}[f(x)] = \sum_x P(x) f(x)$$

连续型：$$E_{x\sim P}[f(x)] = \int p(x) f(x) dx$$

方差：$$Var(f(x)) = E\left[(f(x) - E[f(x)])^2\right]$$

概率论基础

条件概率：

事件A已经发生的条件下，事件B发生的概率： $P(B \mid A)$

随机过程：

设 (Ω, Σ, P) 是一概率空间，对每一个参数 $t \in T$，$X(t,w)$ 是定义在概率空间 (Ω, Σ, P) 上的随机变量

则称随机变量族 $X_T = \{X(t,w); t \in T\}$ 为该概率空间上的一随机过程。 “中科院 孙应飞 随机过程”

马尔科夫过程：

状态满足马尔科夫性的随机过程称为马尔科夫过程

马尔科夫性

马尔科夫性： 系统的下一个状态只与当前状态有关，与以前状态无关。

定义：一个状态St是马尔科夫的，当且仅当：

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

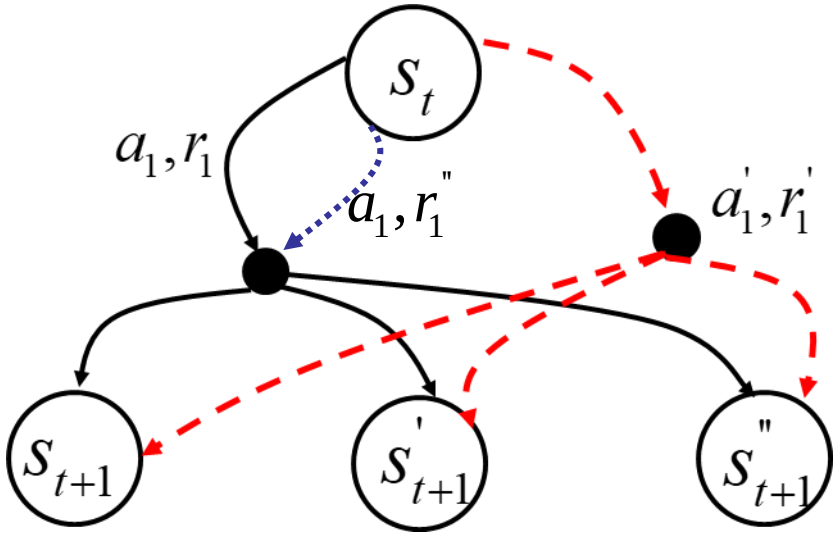
1. 当前状态蕴含所有相关的历史信息
2. 一旦当前状态已知，历史信息将会被抛弃

马尔科夫决策过程（MDP）

MDP：带有决策和回报的马尔科夫过程：

定义：马尔科夫决策过程由元组：(S,A,P,R,γ) 描述

S为有限的状态集，A为有限的行为集，P为状态转移概率，R为回报函数，γ为折扣因子



随机变量 s', r 的分布由条件概率定义：

$$p(s', r \mid s, a) \doteq \Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}$$

该条件概率描述了交互的动力学，是对状态的限制。

状态转移概率：

$$p(s' \mid s, a) \doteq \sum_{r \in \mathcal{R}} p(s', r \mid s, a)$$

回报的期望：

$$r(s,a) \doteq \mathbb{E}\left[R_t \mid S_{t-1} = s, A_{t-1} = a\right] = \sum_{r \in R} r \sum_{s' \in S} p(s', r \mid s, a)$$

马尔科夫决策过程：状态

马尔科夫决策过程是一种很概括和抽象的框架，很多问题都可以建模为马尔科夫决策过程

S 为有限的状态集。

状态：可以是任意我们能够知道的可能有助于做决策的量

DQN 的状态为连续的四帧图像

AlphaGo 的状态为过去八手的双方棋局

The exact architecture, shown schematically in Fig. 1, is as follows. The input to the neural network consists of an $84 \times 84 \times 4$ image produced by the preprocessing map ϕ. The first hidden layer convolves 32 filters of 8×8 with stride 4 with the

Neural network architecture. The input to the neural network is a $19 \times 19 \times 17$ image stack comprising 17 binary feature planes. Eight feature planes, X_t, consist of binary values indicating the presence of the current player's stones ($X_t^i = 1$ if intersection i contains a stone of the player's colour at time-step t; 0 if the intersection is empty, contains an opponent stone, or if $t < 0$). A further 8 feature planes, Y_t, represent the corresponding features for the opponent's stones. The final feature plane, C, represents the colour to play, and has a constant value of either 1 if black is to play or 0 if white is to play. These planes are concatenated together to give input features $s_t = [X_t, Y_t, X_{t-1}, Y_{t-1}, \ldots, X_{t-7}, Y_{t-7}, C]$. History features X_t, Y_t are

状态的表示会严重影响学习效果，状态的表示目前还是一门艺术，而非科学。

本质的问题：三个信号在往前传播和往后传播：状态，动作和回报

马尔科夫决策过程：动作

马尔科夫决策过程是一种很概括和抽象的框架，很多问题都可以建模为马尔科夫决策过程

A为有限的行为集：

动作：任何我们想要学习的需要做的决定

包括宏观动作、微观动作、精神层面的，物质层面的

机器人学习走路：动作可以是关节力矩，还可以是关节角

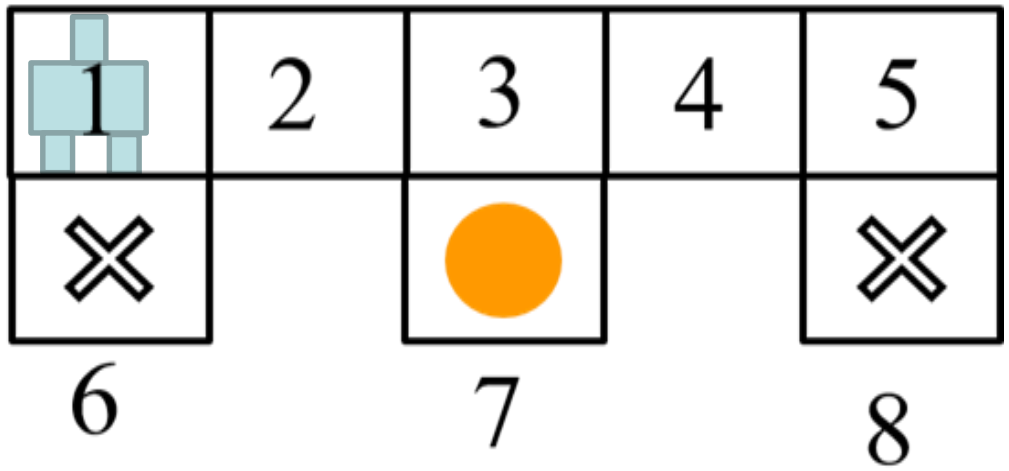
度，甚至可以是电机的电压。

本质的问题：三个信号在往前传播和往后传播：状态，动作和回报

马尔科夫决策过程（MDP）

定义：马尔科夫决策过程是元组：(S, A, P, R, γ)

一个马尔科夫决策过程（很长的交互）：



$$s(1) \xrightarrow[r=0]{a='e'} s(2) \xrightarrow[r=0]{a='e'} s(3) \xrightarrow[r=1]{a='s'} s_T$$

$$s(1) \xrightarrow[r=0]{a='e'} s(2) \xrightarrow[r=0]{a='e'} s(3) \xrightarrow[r=0]{a='e'} s(4) \xrightarrow[r=0]{a='e'} s(5) \xrightarrow[r=-1]{a='s'} s_T$$

在马尔科夫决策过程，智能体学习的目的是最大化总回报

马尔科夫决策过程：立即回报

立即回报的设置：智能体最大化累积回报的时候可以实现我们的最终目标

机器人学习走路：立即回报正比于前向运动的速度

机器人逃离迷宫：每走一步给 -1 以鼓励它尽快逃离迷宫

下棋：赢给+1， 输给-1， 平局给0

智能体的目标：最大化累积回报

马尔科夫决策过程：累积回报

智能体的目标：最大化累积回报

如何定义累积回报？

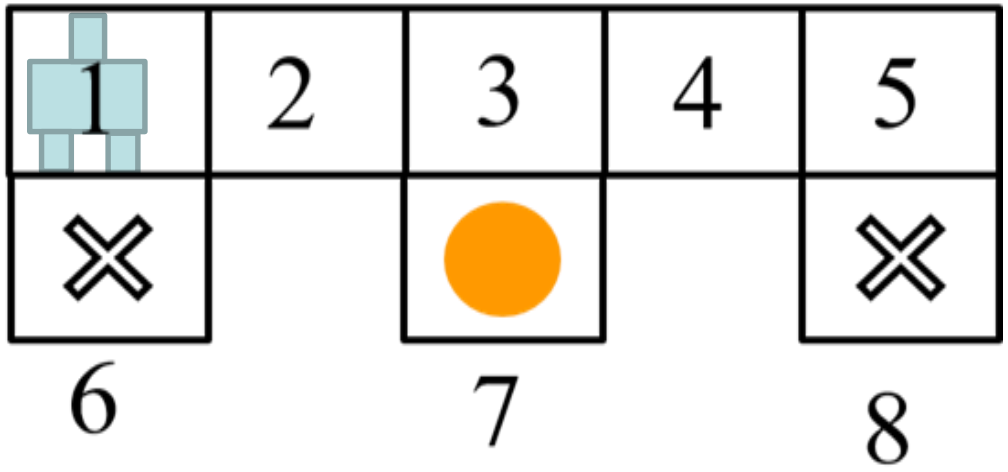
$$s(1) \xrightarrow[r=0]{a='e'} s(2) \xrightarrow[r=0]{a='e'} s(3) \xrightarrow[r=1]{a='s'} s_T$$

回报（return）： $G_t \doteq R_{t+1} + R_{t+2} + \cdots + R_T$

幕（episodes）：一次交互序列

终止状态（terminal state）：终止状态

episodic tasks：带有终止状态的任务

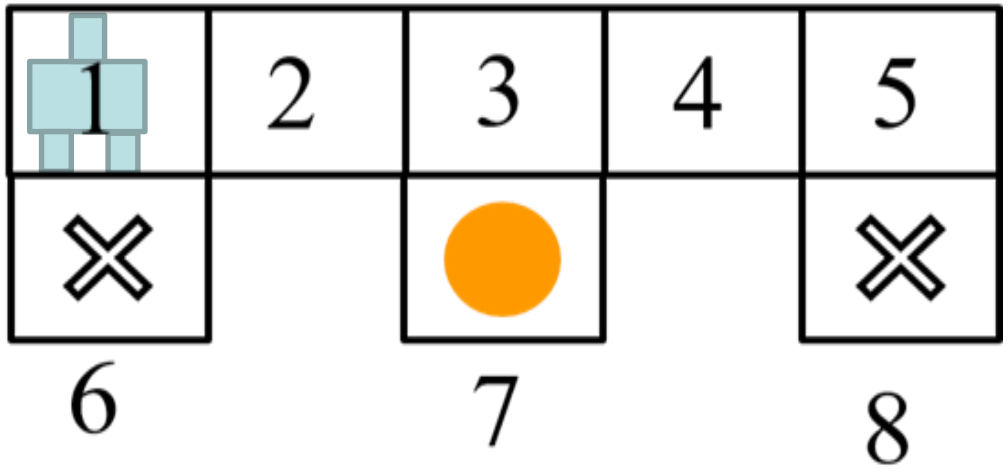


continuing tasks：没有终止状态的，连续进行的任务。如过程控制，机器人运动

马尔科夫决策过程：折扣累积回报

智能体的目标：最大化累积回报

如何定义累积回报？



折扣回报（discounted return）：

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

γ 称为折扣率，并且 $0 \le \gamma \le 1$

γ=0： 智能体是短视的（myopic）， 最大化立即回报

γ 接近1，说明智能体是有远见的（farsighted）， 考虑将来的回报

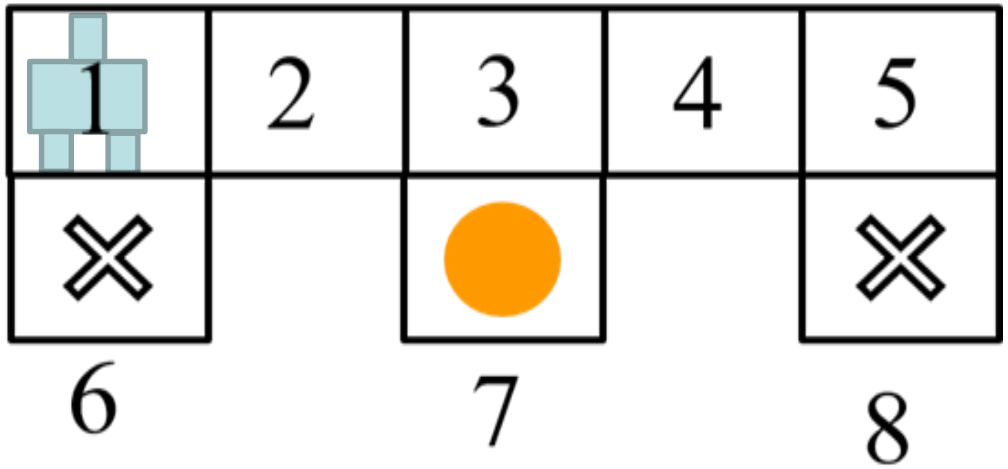
$$G_t \le \frac{1}{1-\gamma} \max R_t$$

马尔科夫决策过程：值函数与策略

$$s(1) \xrightarrow[r=0]{a='e'} s(2) \xrightarrow[r=0]{a='e'} s(3) \xrightarrow[r=1]{a='s'} s_T$$

$$G_t \triangleq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

状态s的好坏：值函数 $\upsilon(s)$（状态的函数）。



值函数跟策略有关，不同的策略对应不同的值函数

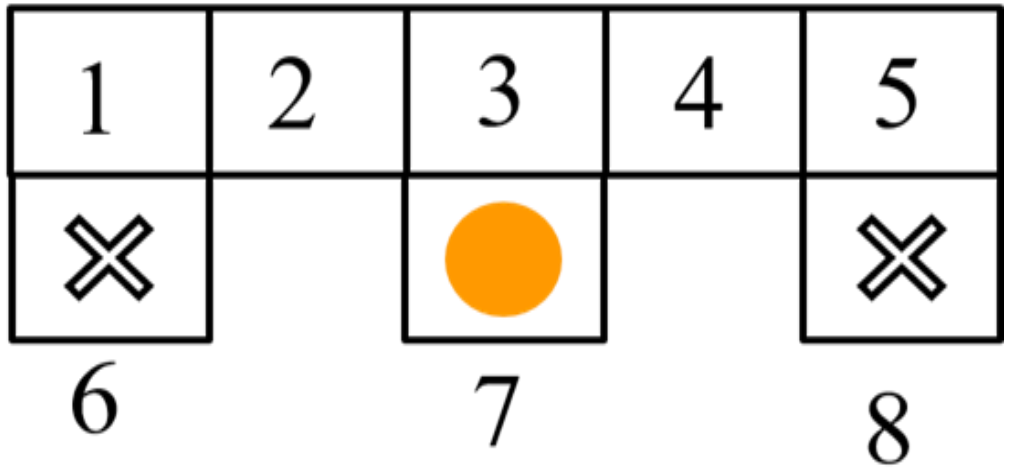
策略的定义：

一个策略 π 是给定状态s时，动作集上的一个分布：

$$\pi(a|s) = p[A_t = a|S_t = s]$$

马尔科夫决策过程：策略的定义

一个策略 π 是给定状态s时，动作集上的一个分布：



状态集： $S=\{1,2,3,4,5,6,7,8\}$

动作集：$A=\{\text{‘e’，‘s’，‘w’，‘n’}\}$

R为立即回报

1. 贪婪策略：

$$\pi_*(a\mid s)=\begin{cases}1 & if \quad a=\arg\max\limits_{a\in A} q_*(s,a)\\ 0 & otherwise\end{cases}$$

2. $\varepsilon-greedy$ 策略：

$$\pi(a|s)\leftarrow\begin{cases}1-\varepsilon+\dfrac{\varepsilon}{|A(s)|} & if\ a=arg\max_a Q(s,a)\\ \dfrac{\varepsilon}{|A(s)|} & if\ a\neq arg\max_a Q(s,a)\end{cases}$$

常用策略举例

3. 高斯策略: $\pi_\theta = \mu_\theta + \varepsilon,\ \varepsilon \sim N(0, \sigma^2)$

4. 玻尔兹曼分布:

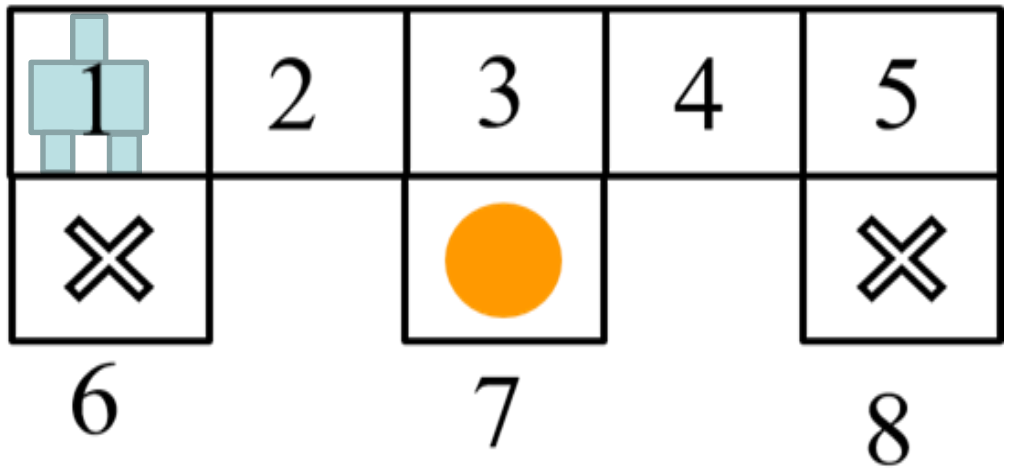
$$\pi_\theta(a \mid s) = \frac{\exp(Q(s,a;\theta))}{\sum_b \exp(Q(s,b;\theta))}$$

马尔科夫决策过程：值函数定义

给定markov链：

$$s(1) \xrightarrow[r=0]{a='e'} s(2) \xrightarrow[r=0]{a='e'} s(3) \xrightarrow[r=1]{a='s'} s_T$$

$$s(1) \xrightarrow[r=0]{a='e'} s(2) \xrightarrow[r=0]{a='e'} s(3) \xrightarrow[r=0]{a='e'} s(4) \xrightarrow[r=0]{a='e'} s(5) \xrightarrow[r=-1]{a='s'} s_T$$



可分别计算累积回报

$$G_t(s) \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

由于状态转移概率和策略的随机性，因此折扣累积回报 $G(s)$ 是随机变量。

值函数的定义：在策略 π 下，状态s的值函数定义为从状态 S 出发，并采用策略 π 的折扣积累回报的期望。

$$\upsilon_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right], \text{ for all } s \in \mathcal{S}$$

马尔科夫决策过程：值函数定义

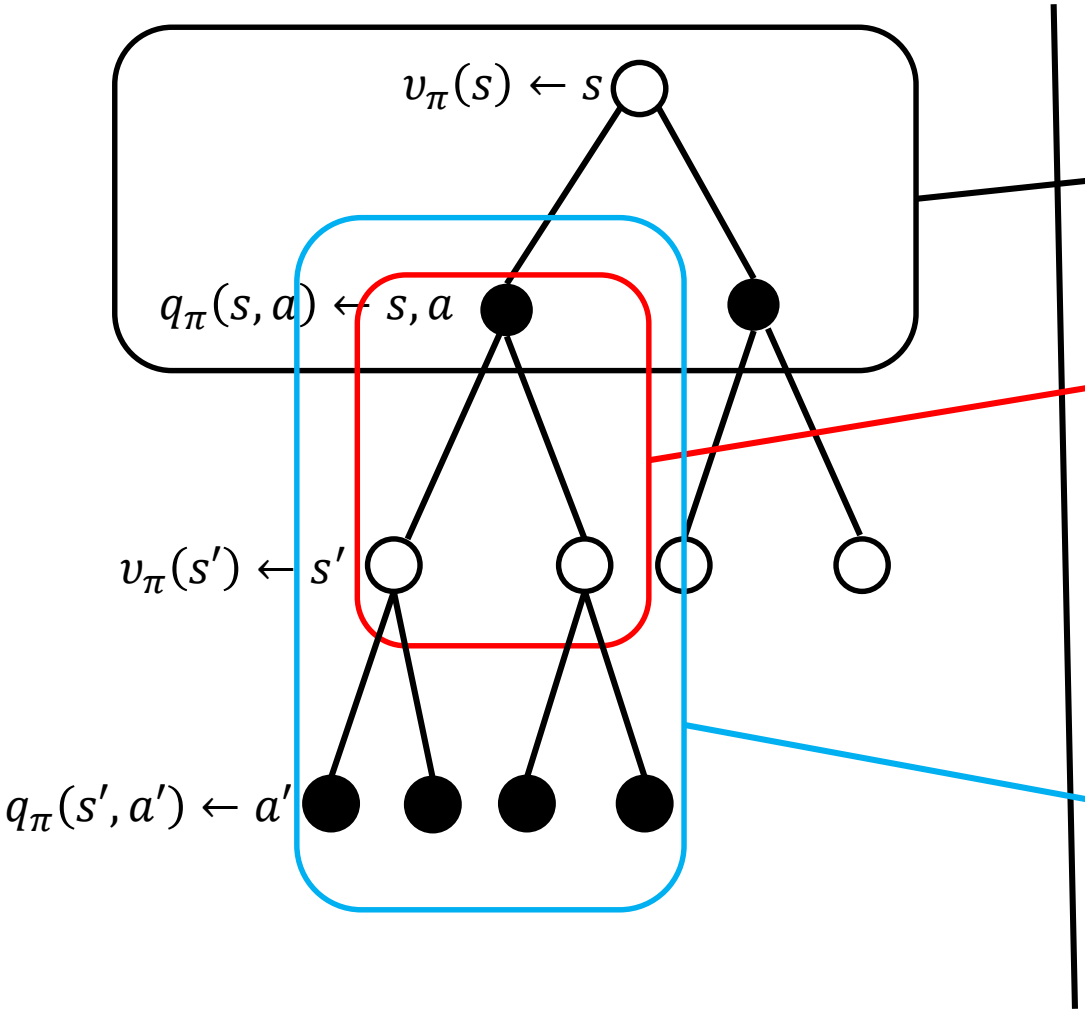
值函数的定义：在策略π下，状态s的值函数定义为从状态 S 出发，并采用策略 π 的折扣积累回报的期望。

$$\upsilon_\pi(s) \triangleq \mathbb{E}_\pi\left[G_t \mid S_t = s\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1} \mid S_t = s\right], \text{for all } s \in \mathcal{S}$$

行为值函数的定义：在策略π下，在状态s，并采取动作a的行为值函数定义为，从状态 S 出发，采用动作a与环境交互，之后采用策略 π所得到的折扣积累回报的期望。

$$q_\pi(s,a) \triangleq \mathbb{E}_\pi\left[G_t \mid S_t = s, A_t = a\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right], \text{for all } s \in \mathcal{S}$$

马尔科夫决策过程：贝尔曼方程



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

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a\, v_\pi(s')$$

$$\Downarrow$$

$$v_\pi(s) = \sum_{a \in A} \pi(a|s) \left(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s') \right)$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')\, q_\pi(s',a')$$

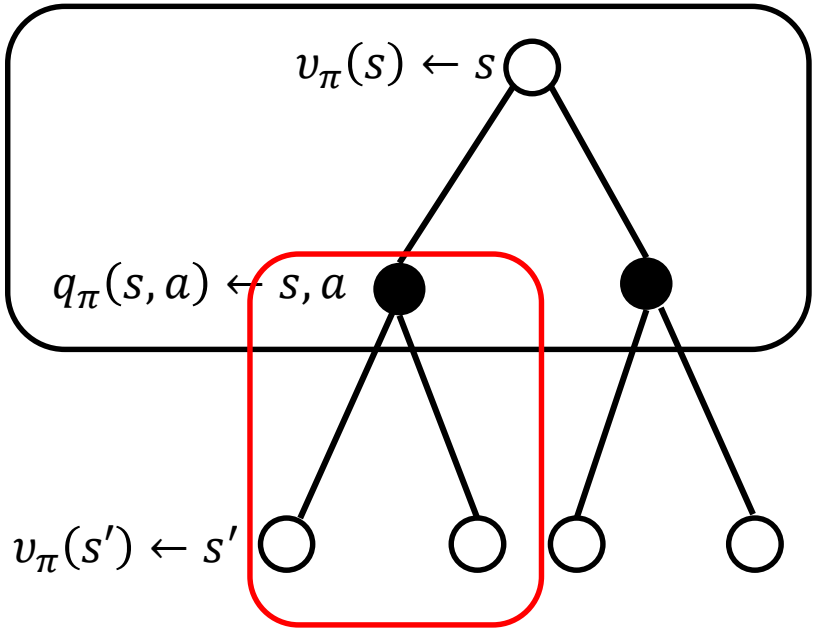
马尔科夫决策过程：贝尔曼方程

值函数的贝尔曼方程

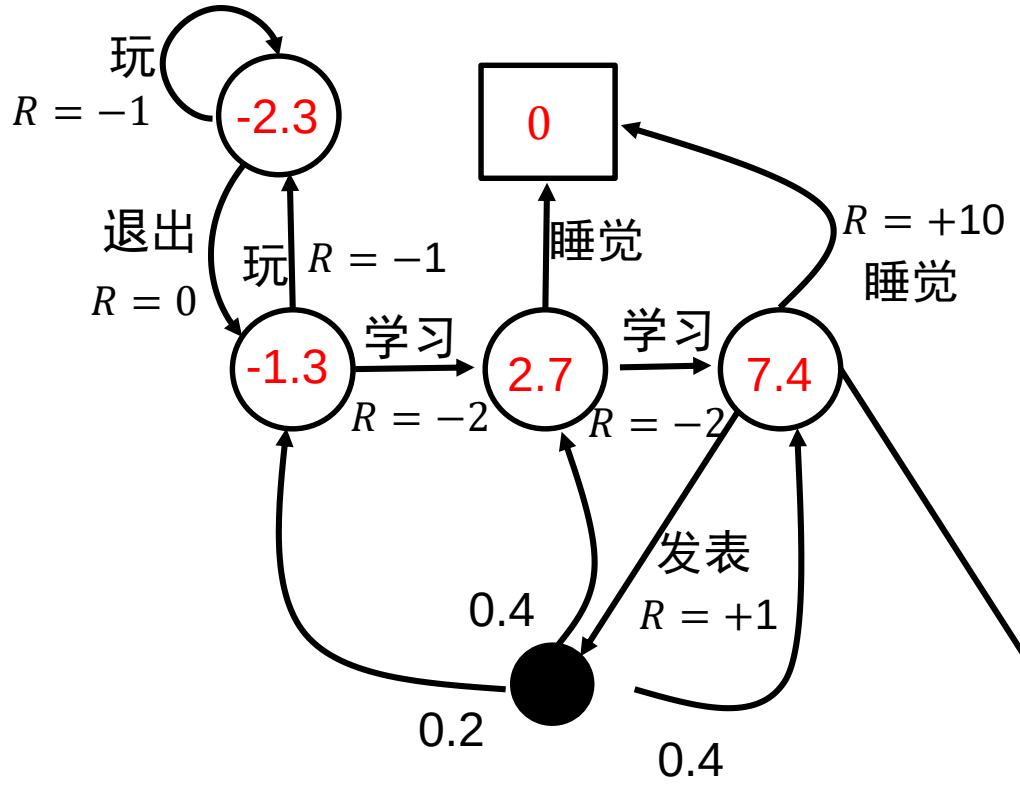
$$v_\pi(s) = \sum_{a \in A} \pi(a|s) \left(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s') \right)$$

回溯：backup计算值函数

后继状态：s’



值函数的具体形式理解



状态集：$S=\{s_1,s_2,s_3,s_4,s_5\}$

动作集：$A=\{$玩，退出，学习，发论文，睡觉$\}$

R为立即回报　$\pi(a|s)=0.5,\gamma=1$

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

$$q_\pi(s,a)=R_s^a+\gamma\sum_{s'\in S}P_{ss'}^a\,v_\pi(s')$$

$$v_\pi(s)=\sum_{a\in A}\pi(a|s)\left(R_s^a+\gamma\sum_{s'\in S}P_{ss'}^a v_\pi(s')\right)$$

$$q_\pi(s,a)=R_s^a+\gamma\sum_{s'\in S}P_{ss'}^a\sum_{a'\in A}\pi(a'|s')\,q_\pi(s',a')$$

$$7.4=0.5(1+0.2*(-1.3)+0.4*2.7+0.4*7.4)+0.5*10$$

最优价值函数和最优状态-动作值函数

最优状态值函数是指 $v^*(s)$ 在所有策略中值最大的值函数，即：$v^*(s) = \max_\pi v_\pi(s)$

最优状态-动作值函数是指 $q^*(s,a)$ 在所有策略中值最大的值函数，即：$q^*(s,a) = \max_\pi q_\pi(s,a)$

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

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a\, v_\pi(s')$$

$$v_\pi(s) = \sum_{a\in A} \pi(a|s)\left(R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a v_\pi(s')\right)$$

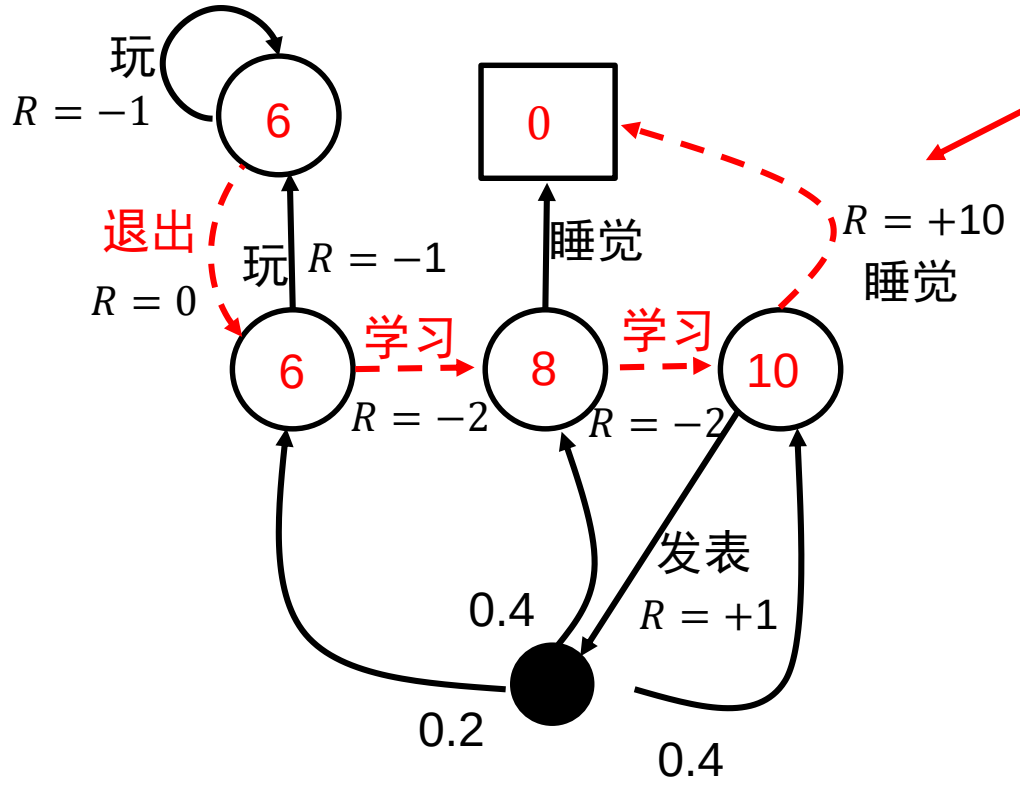
$$q_\pi(s,a) = R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a \sum_{a'\in A} \pi(a'|s')\, q_\pi(s',a')$$

贝尔曼最优化方程：

$$v^*(s) = \max_a R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a\, v_*(s')$$

$$q^*(s,a) = R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a \max_{a'} q^*(s',a')$$

最优策略



状态集：$S=\{s_1,s_2,s_3,s_4,s_5\}$

动作集：$A=$ {玩，退出，学习，发论文，睡觉}

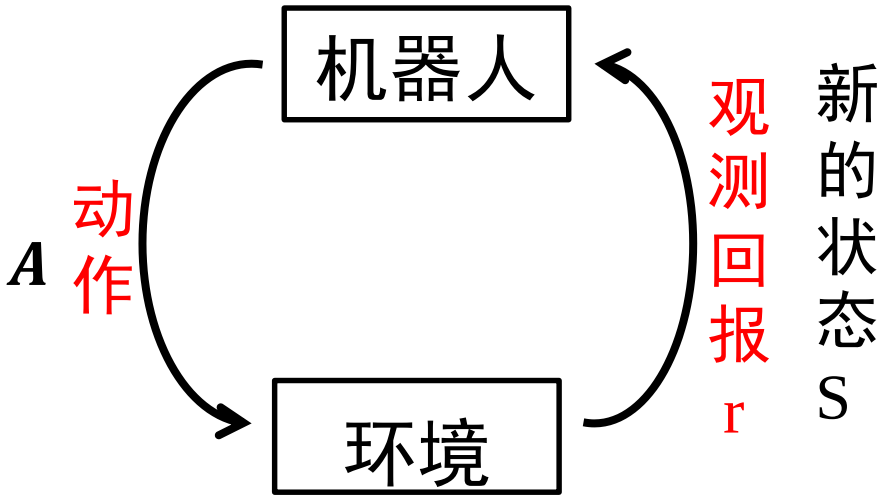
R为立即回报

$$v^*(s)=\max_a R_s^a+\gamma\sum_{s'\in S}P_{ss'}^a v_*(s')$$

若已知最优动作价值函数，最优策略可以通过直接最大化$q_*(s,a)$来决定。

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

强化学习的形式化表示



状态转移概率 $P(S_{t+1}|S_t, a)$

马尔科夫决策问题(MDP):四元组 (S, A, P, r)

策略： $\pi: s \to u$。常采用随机策略：$\pi(u|s)$

累积回报: $R(\tau) = r_T(x_T) + \sum_{t=0}^{T-1} r_t(x_t, u_t)$

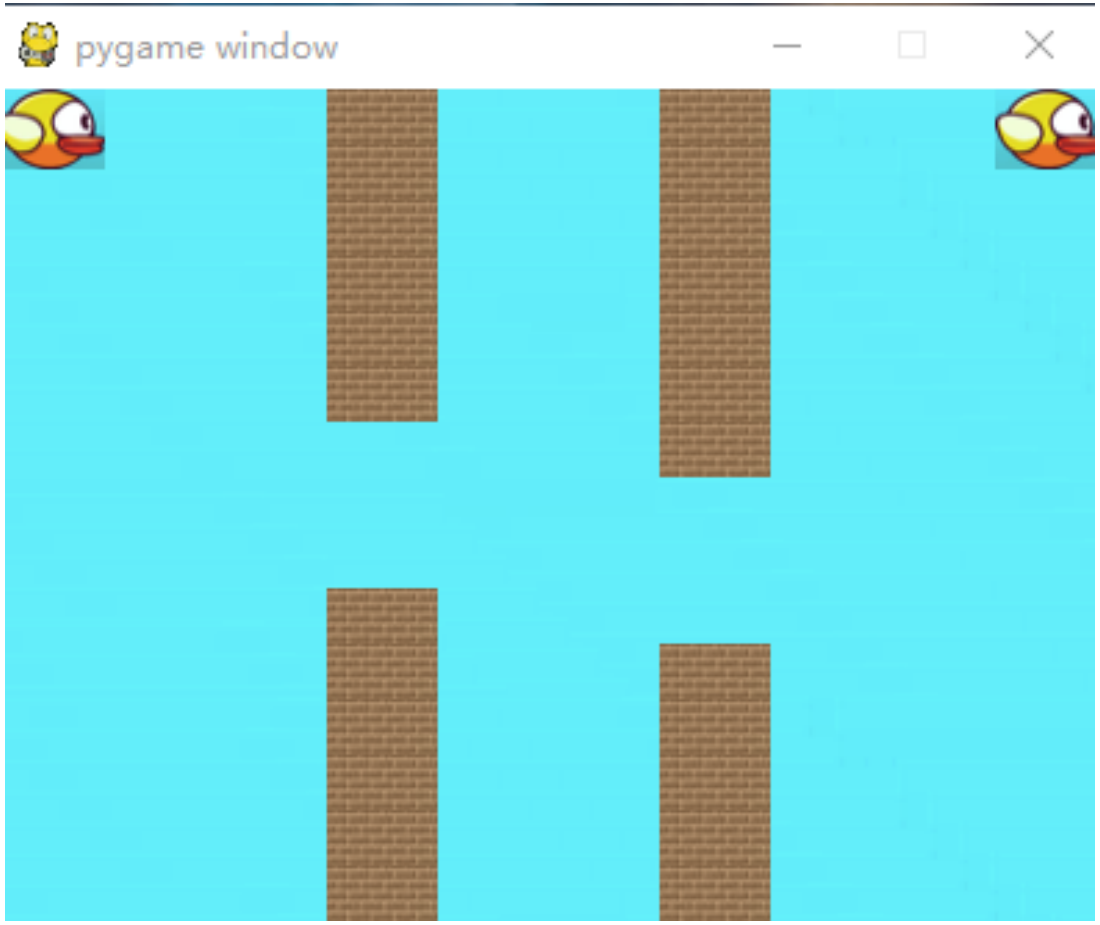
折扣回报: $R = \sum_{t=0}^{\infty} \gamma^t r_t$

强化学习目标： $\max_{\pi} \int R(\tau)\, p_\pi(\tau) d\tau$

值函数

值函数最优时采取的策略是最优的，反过来，策略最优时，值函数最优。

寻找梦中的“她”



找对象

Pygame基础：加载图片

准备工作：

1. 将图片存储到文件夹中
2. 加载系统模块，以便得到图片的目录

```
import os
```

3. 加载 pygame 模块

```
import pygame
```

4. 加载 pygame 模块的函数

```
from pygame.locals import *
```

Step1：获取当前图片所在的目录

```
current_dir = os.path.split(os.path.realpath(__file__))[0]
```

Step2：获取图片的文件名

```
bird_file = current_dir+"/resources/bird.png"
obstacle_file = current_dir+"/resources/obstacle.png"
background_file = current_dir+"/resources/background.png"
```

Step3：下载图片，并转换成surface对象

```
def load_bird_male():
    bird = pygame.image.load(bird_file).convert_alpha()
    return bird
```

Pygame基础：绘图

pygame.cdrom	访问光驱
pygame.cursors	加载光标
pygame.display	访问显示设备
pygame.draw	绘制形状、线和点
pygame.event	管理事件
pygame.font	使用字体
pygame.image	加载和存储图片
pygame.joystick	使用游戏手柄或者 类似的东西
pygame.key	读取键盘按键
pygame.mixer	声音
pygame.mouse	鼠标
pygame.movie	播放视频
pygame.music	播放音频
pygame.overlay	访问高级视频叠加
pygame	就是我们在学的这个东西了……

屏幕显示模块

绘制所需要的形状

主要利用这两个模块

Pygame基础：display 模块

1. 绘制窗口幕布， display.set_mode 函数

```
if self.viewer is None:
    pygame.init()
    self.viewer=pygame.display.set_mode(self.screen_size,0,32)
```

窗口分辨率　窗口性质　色深

2. 在幕布上画图片，blit 函数

```
self.viewer.blit(self.background, (0, 0))
```

图片名字　插入图片的坐标

3. 幕布更新函数：

```
pygame.display.update()
```

第二次作业

1. 阅读 Sutton 书第三章

2. 安装 gym，阅读gym中的代码，写出gym中至少三款游戏的状态、动作、回报、状态转移概率如何设置。