北京航空航天大學
COLLEGE OF SOFTWARE BEIHANG UNIVERSITY 软件学院

人工智能

第14讲：机器学习-强化学习 II

张晶

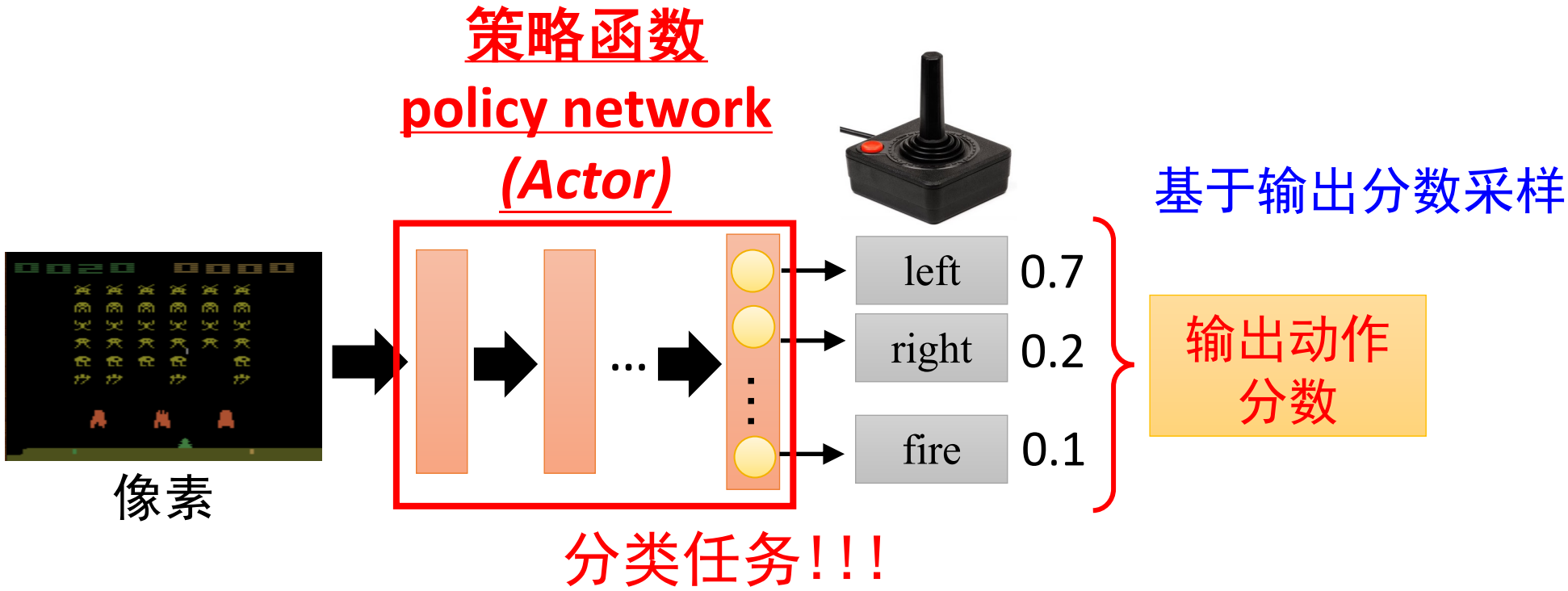
2023年春季

- 参考教材： 吴飞，《人工智能导论：模型与算法》，高等教育出版社
- 在线课程： https://www.icourse163.org/course/ZJU-1003377027?from=searchPage
- 本部分参考：李宏毅，《机器学习》课程，台湾大学

北京航空航天大学 软件学院

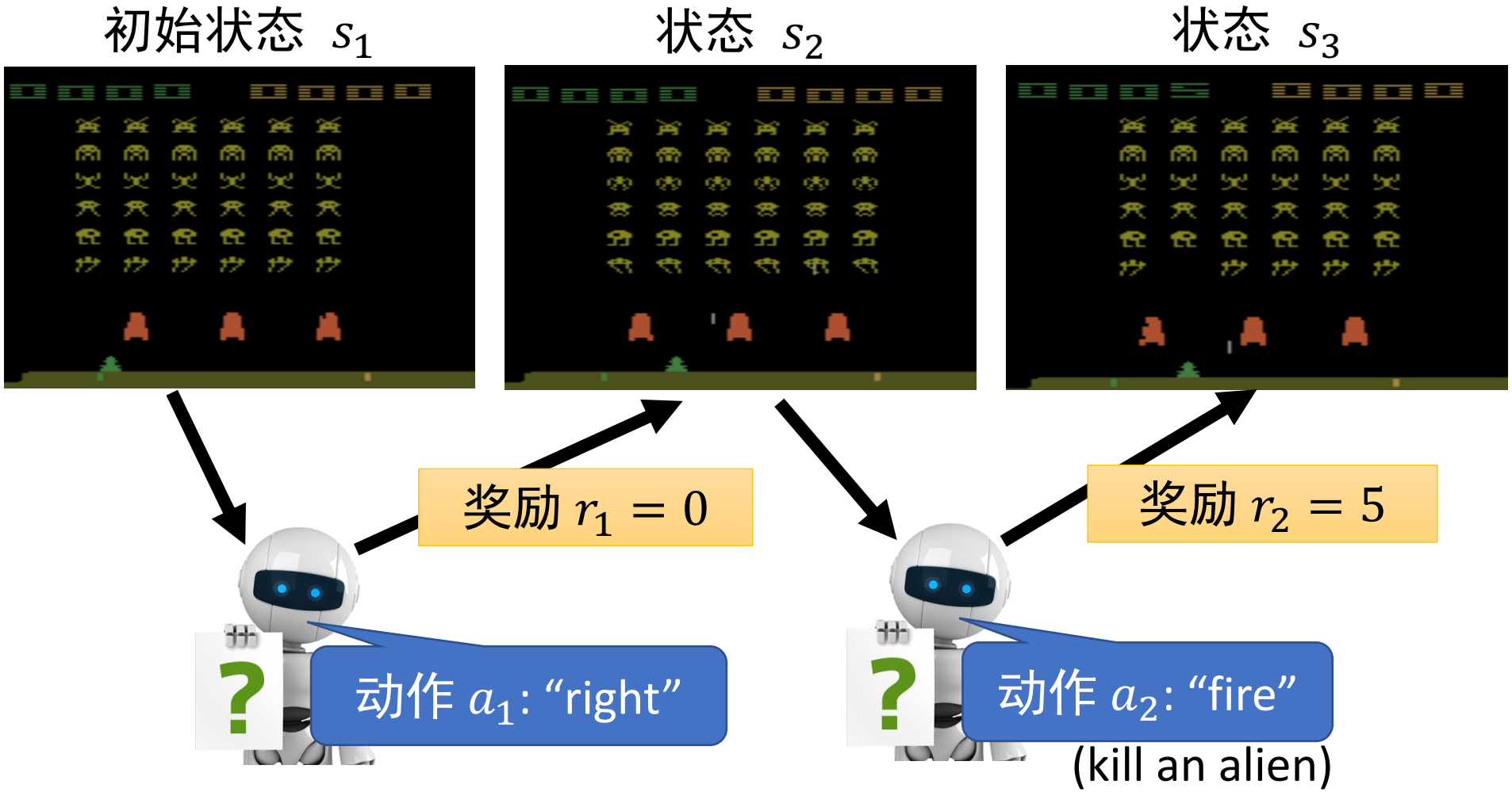
提纲

步骤 1: 定义带有未知参数的函数



- 神经网络的输入：向量或矩阵形式的状态数据
- 神经网络的输出：每个神经元对应的动作

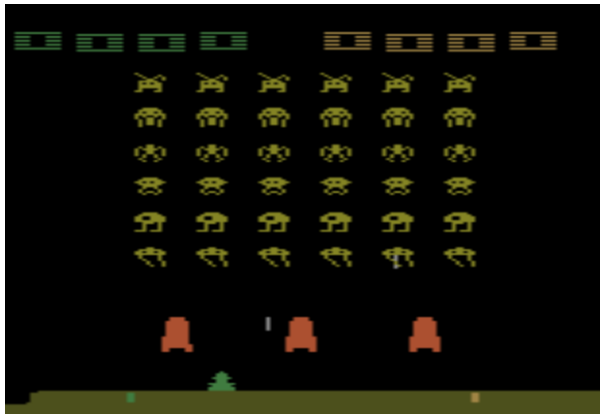
步骤 2: 定义损失函数


初始状态 s_1
状态 s_2
状态 s_3
奖励 $r_1 = 0$
奖励 $r_2 = 5$
?
动作 a_1: “right”
?
动作 a_2: “fire”
(kill an alien)

步骤 2: 定义损失函数

初始状态 s_1

状态 s_2



状态 s_3

After many turns

获得奖励 r_T

动作 a_T

This is an ***episode***（片段）.

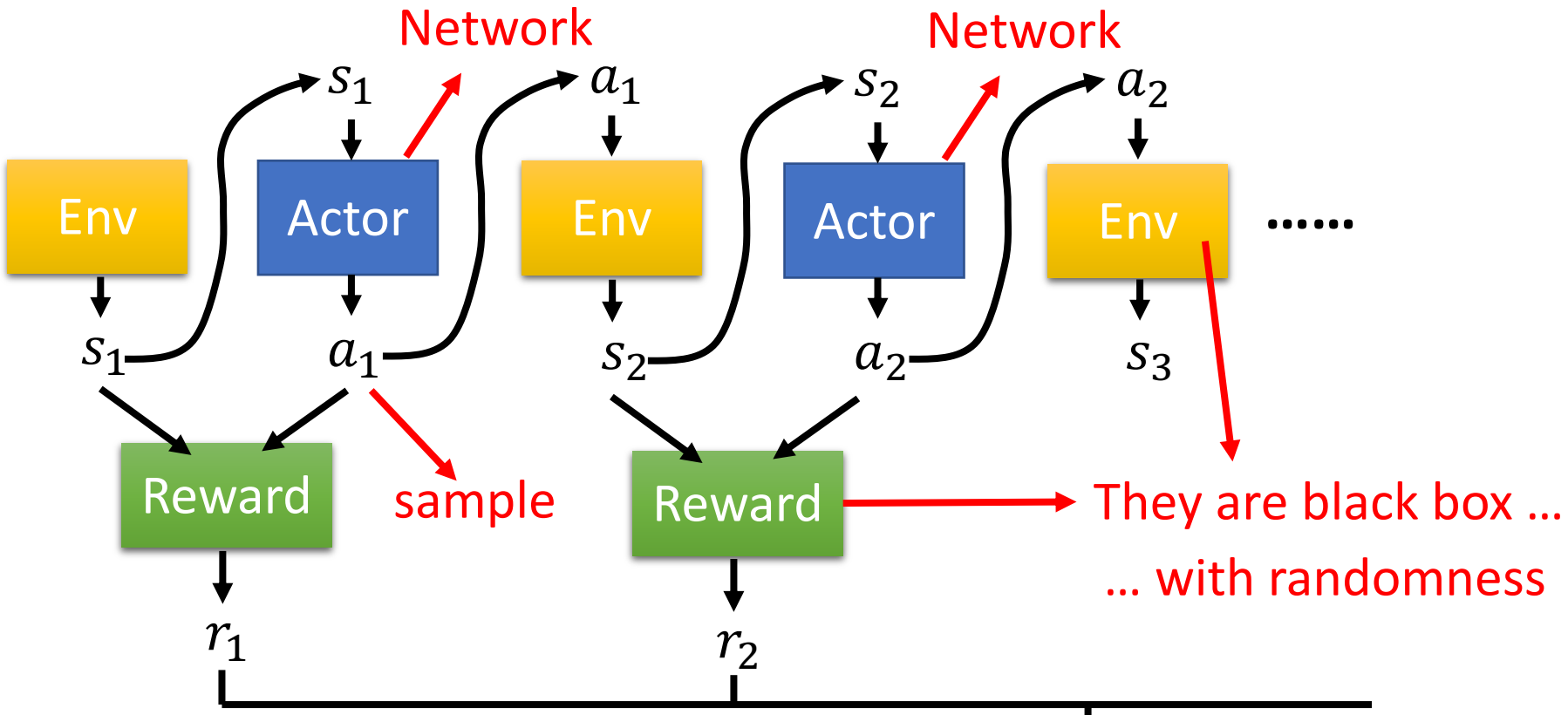
累积奖励（回报）:

$$G = \sum_{t=1}^{T} r_t$$

最大化回报

步骤 3: 优化

Trajectory（轨迹） $\tau = \{s_1, a_1, s_2, a_2, \cdots\}$

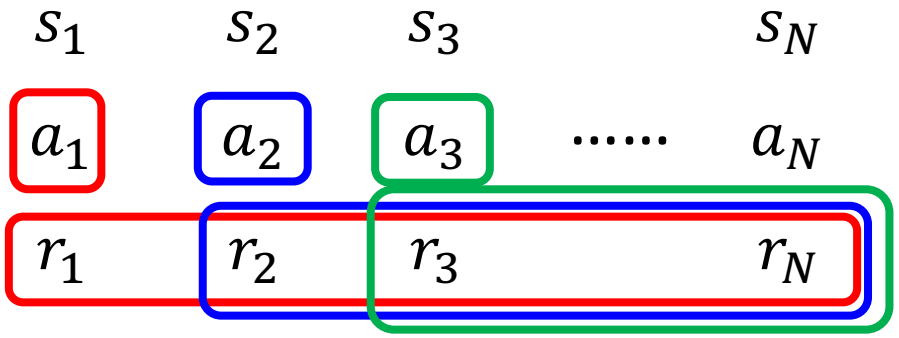


如何进行模型的优化是RL的主要挑战

$$G(\tau) = \sum_{t=1}^{T} r_t$$

c.f. GAN

版本 3



奖励的好坏是“相对的”

如果所有的$r_n \geq 10$,则$r_n = 10$是负奖励...

优势函数
advantage function

Training Data

$$\{s_1, a_1\} \quad A_1 = G'_1 - b$$
$$\{s_2, a_2\} \quad A_2 = G'_2 - b$$
$$\{s_3, a_3\} \quad A_3 = G'_3 - b$$
$$\vdots \qquad \vdots$$
$$\{s_N, a_N\} \quad A_N = G'_N - b$$

减去一个基线值b ???

使得 G'_t 具有正负奖励值

$$G'_t = \sum_{n=t}^{N} \gamma^{n-t} r_n$$

策略梯度 policy gradient

- 初始化策略网络参数π^0
- 迭代训练 $i = 1$ to T
  - 用策略网络π^{i-1} 进行交互
  - 获得训练数据 $\{s_1, a_1\}, \{s_2, a_2\}, \dots, \{s_N, a_N\}$
  - 计算 $A_1, A_2, \dots, A_N$
  - 计算损失L
  - $\pi^i \leftarrow \pi^{i-1} - \eta \nabla L$

数据采集是在训练迭代过程的“for 循环” 内部完成.

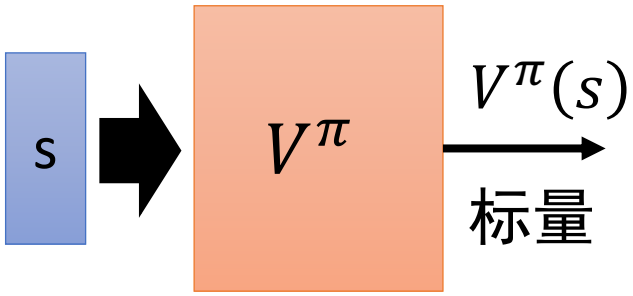
北京航空航天大学 软件学院

提纲

Critic （评判器）

$$G_1' = r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots\cdots$$

- Critic: 对于策略π, 评判状态s (且执行动作a)的好坏
- 价值函数$V^{\pi}(s)$:
  - 使用π时, 观察到状态s后未来获得的折扣累积奖励(discounted *cumulated* reward)的期望。



$V^{\pi}(s)$ 大

$V^{\pi}(s)$ 小

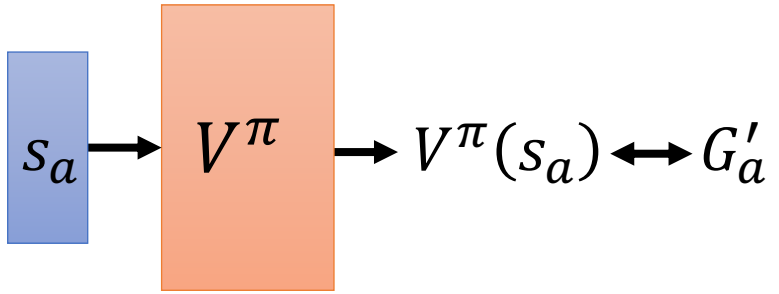
Critic的输出值与当前策略π有关

估计$V^{\pi}(s)$

- **蒙特卡洛法Monte-Carlo (MC)**
  - Critic 观看π与环境进行交互。

观察到s_a后，

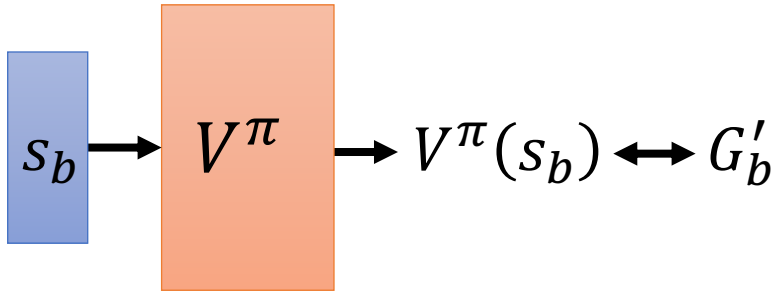
在一个片段(episode)结束后，累积奖励为G'_a



回归问题

观察到s_b后，

在一个片段(episode)结束后，累积奖励为G'_b



回归问题

估计$V^{\pi}(s)$

- **时序差分法Temporal-difference (TD)**

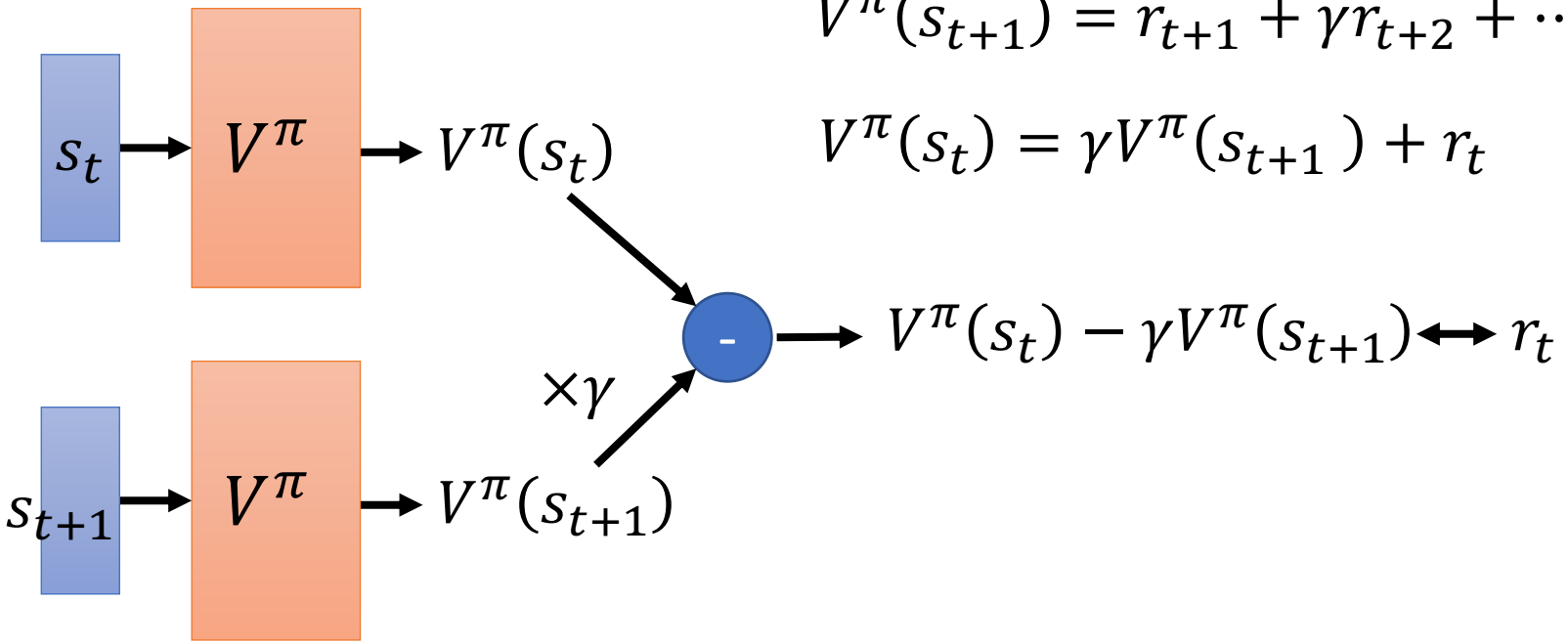
$\cdots s_t, a_t, r_t, s_{t+1} \cdots$

(ignore the expectation here)

$$V^{\pi}(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} \cdots$$

$$V^{\pi}(s_{t+1}) = r_{t+1} + \gamma r_{t+2} + \cdots$$

$$V^{\pi}(s_t) = \gamma V^{\pi}(s_{t+1}) + r_t$$



MC v.s. TD

[Sutton, v2, Example 6.4]

- 假设critic观察到了以下8个片段(episodes)
  - $s_a, r = 0, s_b, r = 0$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 1$, END
  - $s_b, r = 0$, END

(假设 $\gamma = 1$, 并忽略动作.)

$V^{\pi}(s_b) = 3/4$

$V^{\pi}(s_a) =?$ 0? 3/4?

蒙特卡洛法: $V^{\pi}(s_a) = 0$

时序差分法: $V^{\pi}(s_a) = V^{\pi}(s_b) + r$

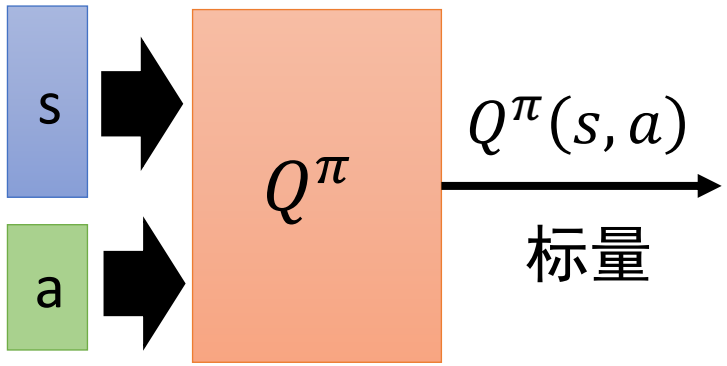
3/4 3/4 0

另一种Critic（评判器）

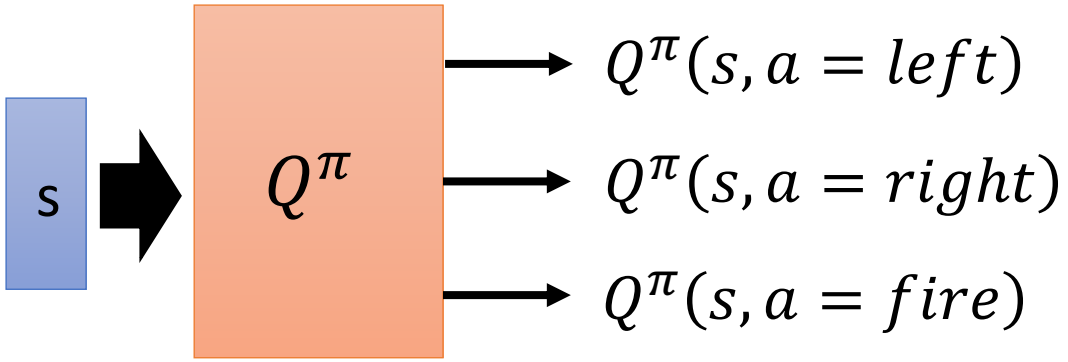
$$G_1' = r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots\cdots$$

- Critic: 对于策略π, 状态s (且执行动作a)的好坏
- 动作-价值函数$Q^{\pi}(s,a)$:
  - 对于策略π, 在状态s下执行动作a后未来获得的折扣累积奖励(discounted *cumulated* reward)的期望。

方法1：



方法2：



仅适用于离散动作

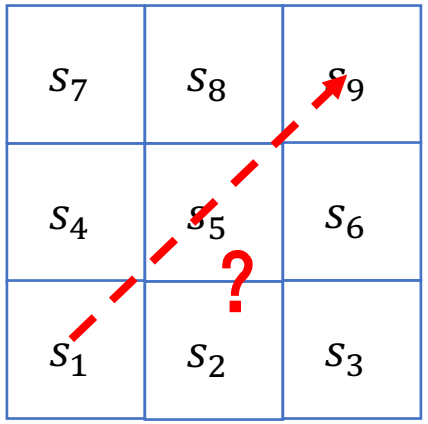
例子：机器人走迷宫

- 以机器人走迷宫为例，详细介绍基于价值的强化学习典型方法

例子：机器人走迷宫

（序列优化）问题：

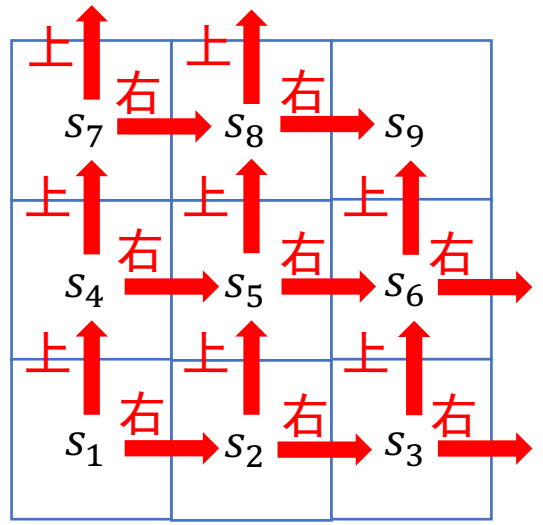
- 在下图网格中，假设有一个机器人位于s_1，其每一步只能向上或向右移动一格，跃出方格会被惩罚（且游戏停止）
- 如何使用强化学习找到一种策略，使机器人从s_1到达s_9？



刻画解该问题的因素

智能主体	迷宫机器人
环境	3×3方格
状态	机器人当前时刻所处方格
动作	每次移动一个方格
奖励	到达s_9时给予奖励；越界时给予惩罚

马尔可夫决策过程（Markov Decision Process）



使用离散马尔可夫过程描述机器人移动问题

- 随机变量序列$\{S_t\}_{t=0,1,2,\cdots}$：S_t表示机器人第t步所在位置（即状态），每个随机变量S_t的取值范围为$S = \{s_1, s_2, \cdots, s_9, s_d\}$
- 动作集合：$A = \{$上，右$\}$
- 状态转移概率$Pr(S_{t+1}|S_t, a_t)$：满足马尔可夫性，其中$a_t \in A$。状态转移如图所示。
- 奖励函数：$R(S_t, a_t, S_{t+1})$
- 衰退系数：$\gamma \in [0, 1]$

下一状态只取决于当前状态

- 动作集合$\boldsymbol{A}$可以是有限的，也可以是无限的
- 状态转移可是确定（deterministic）的，也可以是随机概率性（stochastic）的。
- 确定状态转移相当于发生从$\boldsymbol{S_t}$到$\boldsymbol{S_{t+1}}$的转移概率为1

综合以上信息，可通过$MDP = \{S, A, Pr, R, \gamma\}$来刻画马尔科夫决策过程

机器人走迷宫 - 问题回顾

在机器人移动问题中：状态、行为、衰退系数、起始/终止状态、回报、状态转移概率矩阵的定义如下

$S = \{s_1, s_2, \cdots, s_9, s_d\}$

$A = \{$上，右$\}$

$\gamma = 0.99$

起始状态：$S_0 = s_1$

终止状态：$S_T \in \{s_9, s_d\}$

$$R(S_t, a_t, S_{t+1}) = \begin{cases} 1, \text{如果} S_{t+1} = s_9 \\ -1, \text{如果} S_{t+1} = s_d \\ 0, \text{其他情况} \end{cases}$$

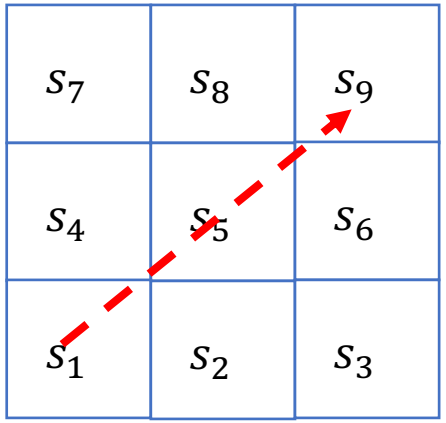
图7.4 机器人寻路问题的状态转移函数

$Pr(S_{t+1}|S_t, a_t = \text{右})$

S_t \ S_{t+1}	s_1	s_2	s_3	...	s_9	s_d
s_1	0	1	0	...	0	0
s_2	0	0	1	...	0	0
s_3	0	0	0	...	0	1
	...			...		...
s_8	0	0	0	...	1	0
s_9	0	0	0	...	0	1

$Pr(S_{t+1}|S_t, a_t = \text{上})$

S_t \ S_{t+1}	s_1	s_4	s_7	...	s_9	s_d
s_1	0	1	0	...	0	0
s_4	0	0	1	...	0	0
s_7	0	0	0	...	0	1
	...			...		...
s_6	0	0	0	...	1	0
s_9	0	0	0	...	0	1



如何从起始状态到终止状态？

强化学习中的策略学习

马尔可夫决策过程$MDP = \{S, A, Pr, R, \gamma\}$对环境进行了描述，那么智能主体如何与环境交互而完成任务？需要进行策略学习

策略函数：

- 策略函数$\pi: S \times A \mapsto [0, 1]$，其中$\pi(s, a)$的值表示在状态$s$下采取动作$a$的概率。
- 策略函数的输出可以是确定的，即给定s情况下，只有一个动作a使得概率$\pi(s, a)$取值为1。对于确定的策略，记为$a = \pi(s)$。
- 如何进行策略学习：一个好的策略是在当前状态下采取了一个行动后，该行动能够在未来收到最大化的回报Return：

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$$

价值函数与动作-价值函数

由马尔可夫性，未来的状态和奖励只与当前状态相关，与t无关。因此t取任意值该等式均成立，如**“逢山开路,遇水搭桥”**。

基于策略的强化学习：

- 直接学习策略函数π，比如策略梯度法

基于价值的强化学习：

为了对策略函数π进行评估，定义如下**critic**

- **价值函数（Value Function）**$V: S \mapsto \mathbb{R}$，其中$V_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s]$，即在第$t$步状态为$s$时，按照策略$\pi$行动后在未来所获得回报值的期望
- **动作-价值函数(Action-Value Function)** $q: S \times A \mapsto \mathbb{R}$，其中$q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$表示在第$t$步状态为$s$时，按照策略$\pi$采取动作$a$后，在未来所获得回报值的期望

这样，策略学习转换为如下优化问题：

寻找一个最优策略π^*，对任意$s \in S$使得$V_{\pi^*}(s)$值最大

价值函数与动作-价值函数的关系

- 价值函数（Value Function） $V_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s]$
- 动作-价值函数(Action-Value Function) $q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]$

$$
\begin{aligned}
&V_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s] \\
&= \mathbb{E}_{a\sim\pi(s,\cdot)}\left[\mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]\right] \\
&= \sum_{a\in A} \underbrace{\pi(s,a)}_{\text{采取动作}a\text{的概率}} \times \underbrace{q_\pi(s,a)}_{\text{采取动作}a\text{后带来的回报期望}} \\
&= \sum_{a\in A} \pi(s,a) q_\pi(s,a)
\end{aligned}
$$

价值函数与动作-价值函数的关系

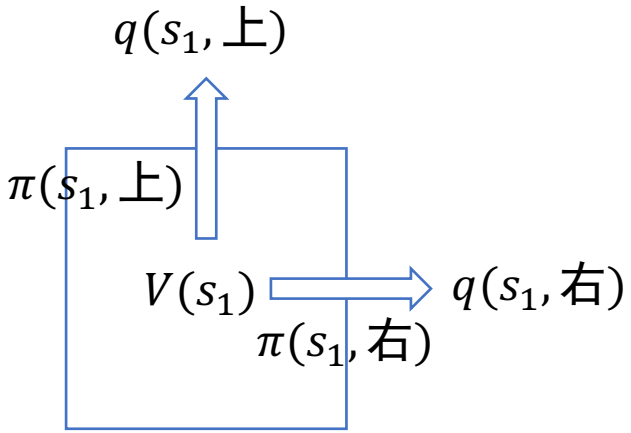
- 价值函数（Value Function） $V_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s]$
- 动作-价值函数(Action-Value Function) $q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]$

$$q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]$$
$$= \mathbb{E}_{s' \sim P(\cdot|s,a)}\left[R(s,a,s') + \gamma \mathbb{E}_\pi[R_{t+2} + \gamma R_{t+3} + \cdots | S_{t+1} = s']\right]$$
$$= \sum_{s' \in \boldsymbol{S}} \underbrace{P(s'|s,a)}_{\text{在状态s采取行动a进入状态s'的概率}} \times [\underbrace{R(s,a,s')}_{\text{在s采取a进入s'得到的回报}} + \gamma \times \underbrace{V_\pi(s')}_{\text{在s'获得的回报期望}}]$$
$$= \sum_{s' \in \boldsymbol{S}} P(s'|s,a)\,[R(s,a,s') + \gamma V_\pi(s')]$$

价值函数与动作-价值函数的关系：以状态s_1的计算为例

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

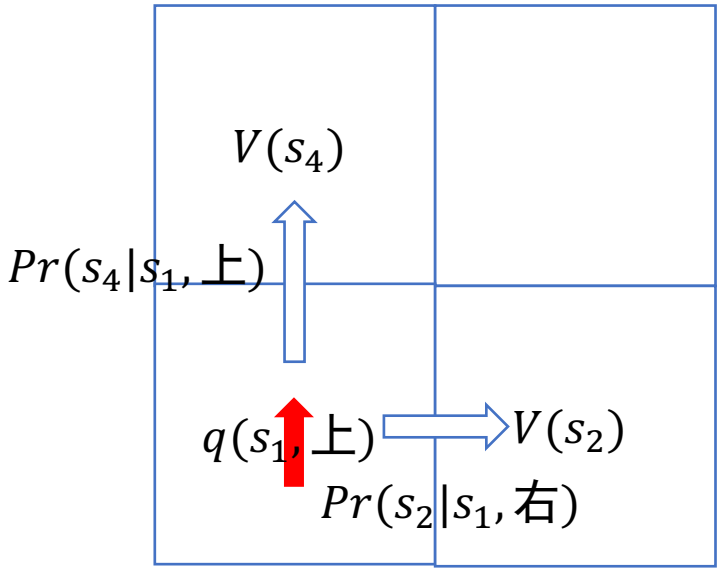
$$V_\pi(s_1) = \pi(s_1, 上) q_\pi(s_1, 上) + \pi(s_1, 右) q_\pi(s_1, 右)$$



不同动作下的回报累加

$$q_\pi(s,a) = \sum_{s' \in S} Pr(s'|s,a) [R(s,a,s') + \gamma V_\pi(s')]$$

$$q_\pi(s_1, 上) = Pr(s_4|s_1, 上)[R(s_1, 上, s_4) + \gamma V_\pi(s_4)]$$



动作确定时状态转移后的回报结果

贝尔曼方程 (Bellman Equation)：递推关系

贝尔曼方程（Bellman Equation）也被称作动态规划方程（Dynamic Programming Equation），由理查德·贝尔曼（Richard Bellman）提出。

- 价值函数（Value Function） $V_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s]$
- 动作-价值函数(Action-Value Function) $q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]$

$$V_\pi(s) = \sum_{a \in A} \pi(s,a) \sum_{s' \in \boldsymbol{S}} P(s'|s,a)\,[R(s,a,s') + \gamma V_\pi(s')]$$
$$= \mathbb{E}_{a \sim \pi(s,\cdot)} \mathbb{E}_{s' \sim P(\cdot|s,a)} [R(s,a,s') + \gamma V_\pi(s')]$$

价值函数取值与时间没有关系，只与策略π、在策略π下从某个状态转移到其后续状态所取得的回报以及在后续所得回报有关。

贝尔曼方程 (Bellman Equation)：递推关系

贝尔曼方程（Bellman Equation）也被称作动态规划方程（Dynamic Programming Equation），由理查德·贝尔曼（Richard Bellman）提出。

- 价值函数（Value Function） $V_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s]$
- 动作-价值函数(Action-Value Function) $q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s, A_t = a]$

$$q_\pi(s,a) = \sum_{s' \in \boldsymbol{S}} P(s'|s,a)\left[R(s,a,s') + \gamma \sum_{a' \in A} \pi(s',a') q_\pi(s',a')\right]$$
$$= \mathbb{E}_{s' \sim P(\cdot|s,a)}[R(s,a,s') + \gamma \mathbb{E}_{a' \sim \pi(s',\cdot)}[q_\pi(s',a')]]$$

动作-价值函数取值同样与时间没有关系，而是与瞬时奖励和下一步的状态和动作有关。

贝尔曼方程 (Bellman Equation)：递推关系

价值函数的贝尔曼方程

$$V_\pi(s) = \mathbb{E}_{a\sim\pi(s,\cdot)}\ \mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma V_\pi(s')]$$

动作-价值函数的贝尔曼方程

$$q_\pi(s,a) = \mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma \mathbb{E}_{a'\sim\pi(s',\cdot)}[q_\pi(s',a')]]$$

- 贝尔曼方程描述了价值函数或动作-价值函数的递推关系，是研究强化学习问题的重要手段。
- 其中价值函数的贝尔曼方程描述了当前状态价值函数和其后续状态价值函数之间的关系，即当前状态价值函数等于瞬时奖励的期望加上后续状态的（折扣）价值函数的期望。
- 而动作-价值函数的贝尔曼方程描述了当前动作-价值函数和其后续动作-价值函数之间的关系，即当前状态下的动作-价值函数等于瞬时奖励的期望加上后续状态的（折扣）动作-价值函数的期望。

北京航空航天大學 软件学院

贝尔曼方程 (Bellman Equation)：递推关系

价值函数的贝尔曼方程

$$V_\pi(s) = \mathbb{E}_{a\sim\pi(s,\cdot)}\ \mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma V_\pi(s')]$$

动作-价值函数的贝尔曼方程

$$q_\pi(s,a) = \mathbb{E}_{s'\sim P(\cdot|s,a)}[R(s,a,s') + \gamma \mathbb{E}_{a'\sim\pi(s',\cdot)}[q_\pi(s',a')]]$$

- 在实际中，需要计算得到最优策略以指导智能体在当前状态如何选择一个可获得最大回报的动作。
- 求解最优策略的一种方法就是去求解最优的价值函数或最优的动作-价值函数（即基于价值方法，value-based approach）。
- 一旦找到了最优的价值函数或动作-价值函数，自然而然也就是找到最优策略。

策略迭代的基本模式

为了求解最优策略π^*，上图展示了一种思路：从一个任意的策略开始，首先计算该策略下价值函数（或动作-价值函数），然后根据价值函数调整改进策略使其更优，不断迭代这个过程直到策略收敛。

- 通过策略计算价值函数的过程叫做策略评估（policy evaluation）
- 通过价值函数优化策略的过程叫做策略优化（policy improvement）
- 策略评估和策略优化交替进行的强化学习求解方法叫做通用策略迭代（Generalized Policy Iteration，GPI）。

强化学习中的策略优化

策略优化定理：

对于确定的策略π和π'，如果对于任意状态$s \in S$

$$q_\pi(s, \pi'(s)) \geq q_\pi(s, \pi(s))$$

那么对于任意状态$s \in S$，有

$$V_{\pi'}(s) \geq V_\pi(s)$$

即策略π'不比π差

注意，不等式左侧的含义是只在当前这一步将动作修改为$\pi'(s)$，未来的动作仍然按照π的指导进行

在讨论如何优化策略之前，首先需要明确什么是“更好”的策略。分别给出π和π'两个策略，如果对于任意状态$s \in S$，有$V_\pi(s) \leq V_{\pi'}(s)$，那么可以认为策略$\pi'$不比策略$\pi$差，可见“更优”策略是一个偏序关系。

强化学习中的策略优化

给定当前策略π、价值函数V_π和行动-价值函数q_π时，可如下构造新的策略π'，只要 π'满足如下条件：

$$\pi'(s) = \text{argmax}_a q_\pi(s,a) \quad （对于任意s \in S）$$

π'便是对π的一个改进。于是对于任意$s \in S$，有

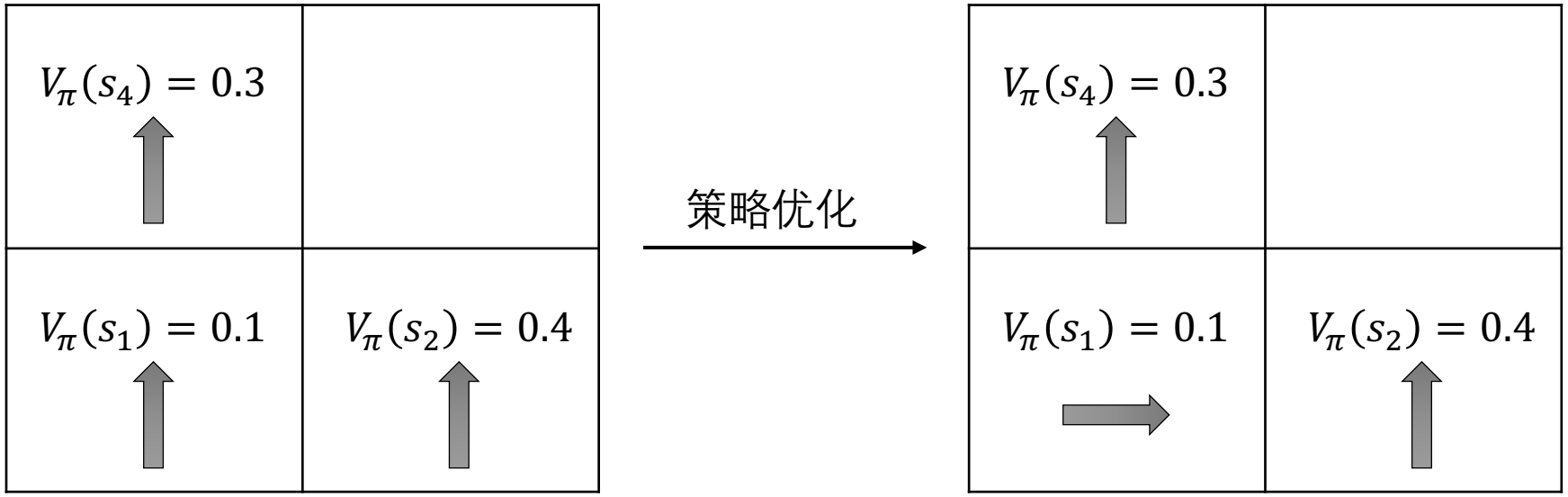
$$q_\pi\big(s,\pi'(s)\big) = q_\pi\big(s, argmax_a\ q_\pi(s,a)\big)$$

$$= \max_a q_\pi(s,a)$$

$$\geq q_\pi\big(s,\pi(s)\big) = V_\pi(s)$$

强化学习中的策略优化：机器人寻路问题为例子

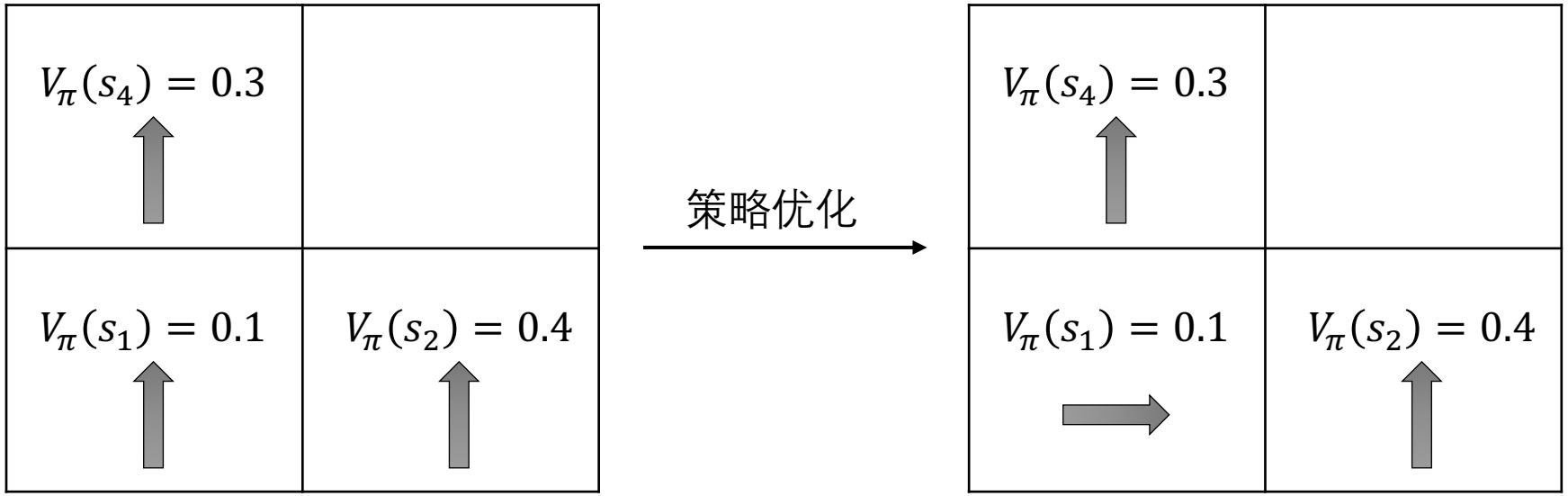
图7.8 对状态s_1所采取的行动进行策略优化的示意图



- 图7.8给出了机器人寻路问题的原有策略及其对应价值函数。
- 在图中，根据原有策略，智能体位于状态s_1的价值函数取值为0.1。
- 当智能体位于s_1状态时，智能体在原有策略指引下将选择向上移动一个方格的行动，从状态s_1进入状态s_4，状态s_4的价值函数取值为0.3。原有策略所给出的其他信息，可从图中周知。

强化学习中的策略优化：机器人寻路问题为例子

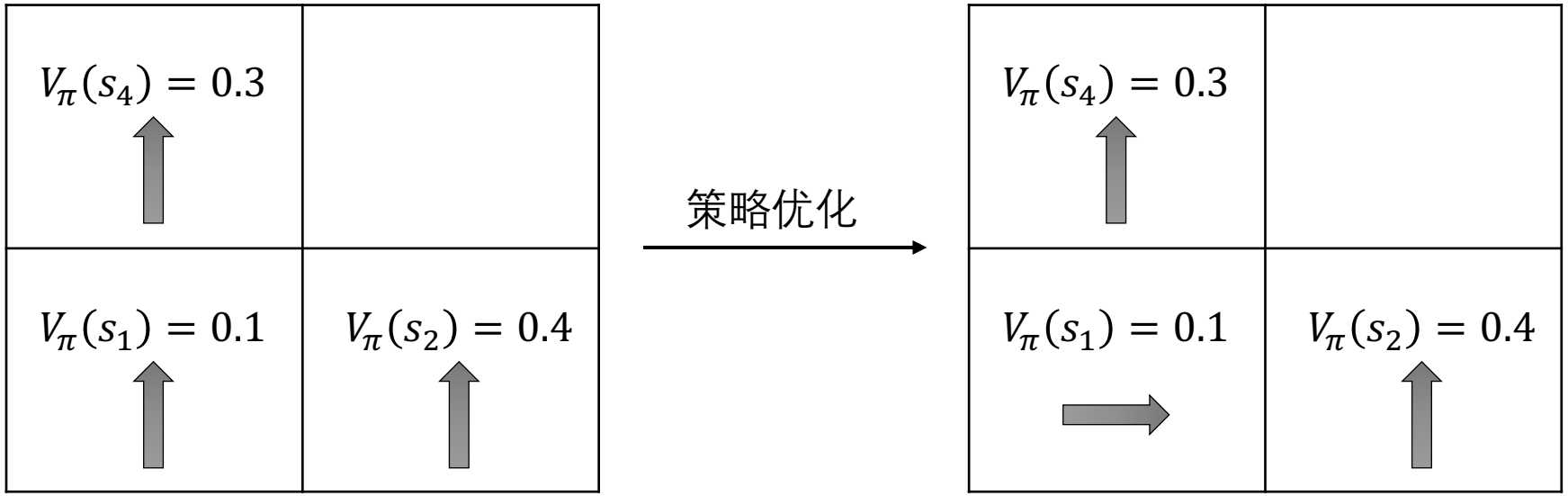
图7.8 对状态s_1所采取的行动进行策略优化的示意图



- 下面来看智能体如何通过策略优化来改变在状态s_1所采取的行动。
- 由于智能体在状态s_1能够采取“向上移动一个方格”或“向右移动一个方格”两个行动中的一个。
- 首先计算状态s_1选择“向上移动一个方格”后所得动作-价值函数取值。
- 接着计算状态s_1选择“向右移动一个方格”后所得动作-价值函数取值。

强化学习中的策略优化：机器人寻路问题为例子

图7.8 对状态s_1所采取的行动进行策略优化的示意图



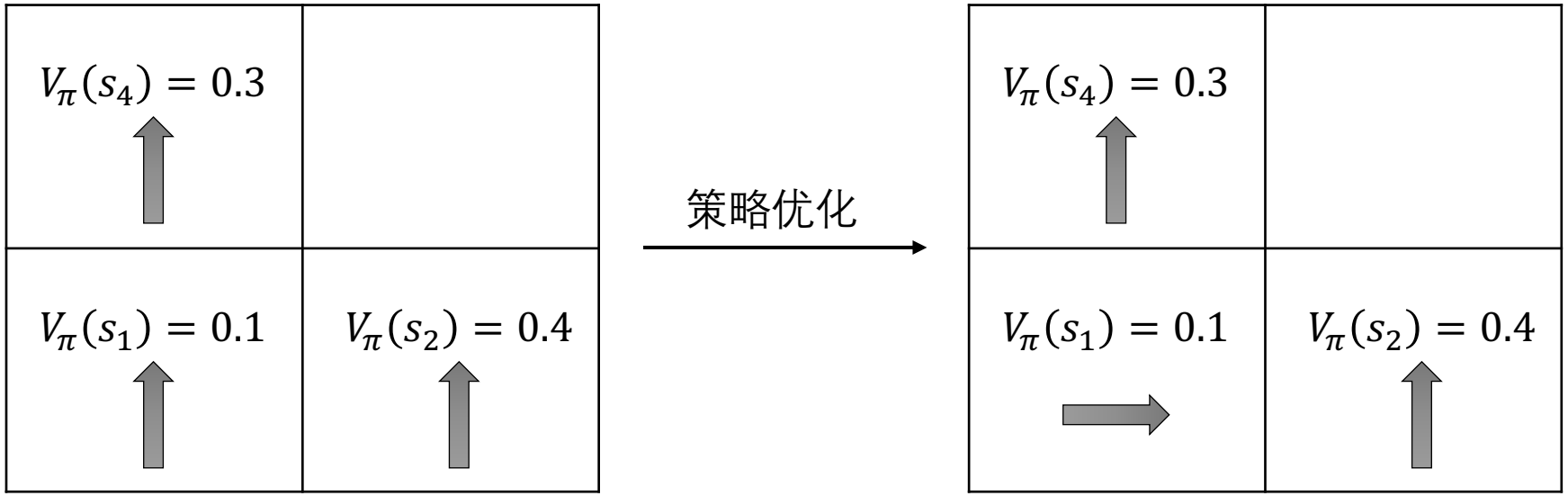
$$q_\pi(s_1, 上) = \sum_{s' \in S} \mathrm{P}(s'|s_1, 上)\,[R(s_1, 上, s') + \gamma V_\pi(s')]$$
$$= 1\times(0 + 0.99\times0.3) + 0\times\cdots = 0.297$$

$$q_\pi(s_1, 右) = \sum_{s' \in S} \mathrm{P}(s'|s_1, 右)\,[R(s_1, 右, s') + \gamma V_\pi(s')]$$
$$= 1\times(0 + 0.99\times0.4) + 0\times\cdots = 0.396$$

$$R(S_t, a_t, S_{t+1}) = \begin{cases} 1, 如果S_{t+1} = s_9 \\ -1, 如果S_{t+1} = s_d \\ 0, 其他情况 \end{cases}$$

强化学习中的策略优化：机器人寻路问题为例子

图7.8 对状态s_1所采取的行动进行策略优化的示意图



$$q_\pi(s_1, 右) == 0.396 > q_\pi(s_1, 上) = 0.297$$

可见，智能体在状态s_1选择“向上移动一个方格”行动所得回报$q_\pi(s_1, 上)$值为0.297、选择“向右移动一个方格”行动所得回报$q_\pi(s_1, 右)$值为0.396。显然，智能体在状态s_1应该选择“向右移动一个方格”行动，这样能够获得更大的回报。于是，经过策略优化后，状态s_1处的新策略为$\pi'(s_1) = argmax_a q_\pi(s, a) = 右$，则将$s_1$处的策略从“上”更新为“右”。其他状态的情况可用类似方法计算得到。

北京航空航天大學 软件学院

强化学习中的策略评估方法

通过策略优化得到了新的策略，需要对新策略进行评估，以更新价值函数或动作价值函数，进而进一步优化策略。

假定当前策略为π，策略评估指的是根据策略π来计算相应的价值函数V_π或动作-价值函数q_π。三种常见的策略评估方法包括

- 动态规划 （**Model-based** Reinforcement Learning，状态转移概率已知）
- 蒙特卡洛采样（**Model-free** Reinforcement Learning）
- 时序差分（Temporal Difference）（**Model-free** Reinforcement Learning）

强化学习中的策略评估：**动态规划**

基于动态规划的价值函数更新：使用迭代的方法求解贝尔曼方程组

初始化V_π函数
循环
　　枚举$s \in S$
　　$V_\pi(s) \leftarrow \sum_{a\in A}\pi(s,a)\sum_{s'\in S}Pr(s'|s,a)\left[R(s,a,s')+\gamma V_\pi(s')\right]$

直到V_π收敛

更新$V_\pi(s_1)$的值：

$q_\pi(s_1,\text{上}) = 1\times(0+0.99\times0.3)+0\times(0+0.99\times0.4)+\ldots = 0.297$

$V_\pi(s_1) = 1\times q_\pi(s_1,\text{上})+0\times q_\pi(s_1,\text{右}) = 0.297$

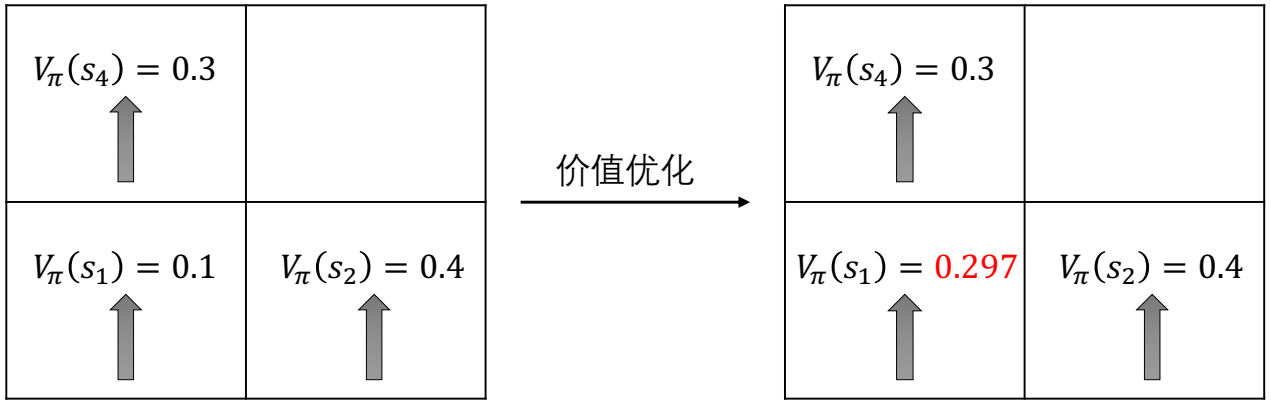
强化学习中的策略评估：**动态规划**

基于动态规划的价值函数更新：使用迭代的方法求解贝尔曼方程组

更新$V_\pi(s_1)$的值：

$$q_\pi(s_1, 上) = 1\times(0 + 0.99\times0.3) + 0\times(0 + 0.99\times0.4) + ... = 0.297$$

$$V_\pi(s_1) = 1\times q_\pi(s_1, 上) + 0\times q_\pi(s_1, 右) = 0.297$$



- **动态规划法的缺点：**

1) 智能主体需要事先知道状态转移概率;

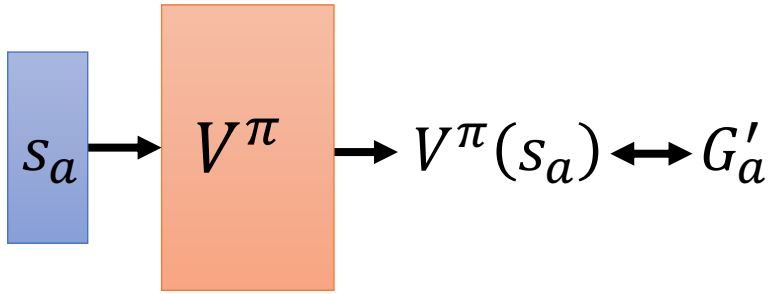
2) 无法处理状态集合大小无限的情况

强化学习中的策略评估：**蒙特卡洛采样** - 估计$V^{\pi}(s)$

- **蒙特卡洛法Monte-Carlo (MC)**
  - Critic 观看π与环境进行交互。

观察到s_a后，

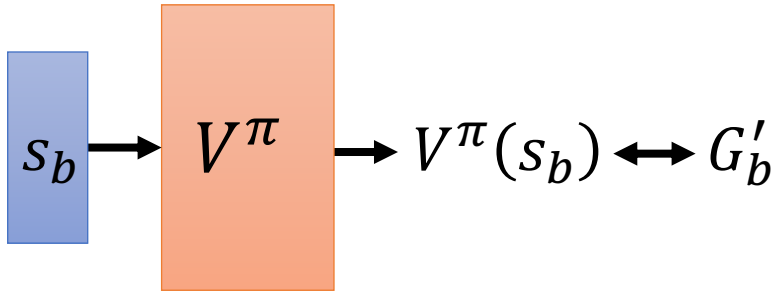
在一个片段(episode)结束后，累积奖励为G'_a



回归问题

观察到s_b后，

在一个片段(episode)结束后，累积奖励为G'_b



回归问题

强化学习中的策略评估：蒙特卡洛采样

基于蒙特卡洛采样的价值函数更新

选择不同的起始状态，按照当前策略π采样若干轨迹，记它们的集合为D

枚举$s \in S$

计算D中s每次出现时对应的回报$G_1, G_2, \cdots, G_k$

$V_\pi(s) \leftarrow \frac{1}{k}\sum_{i=1}^{k} G_i$

假设按照当前策略可以得到以下两条轨迹

$$(s_1, s_4, s_7, s_8, s_9)$$

$$(s_1, s_2, s_3, s_d)$$

s_1对应的回报值分别为

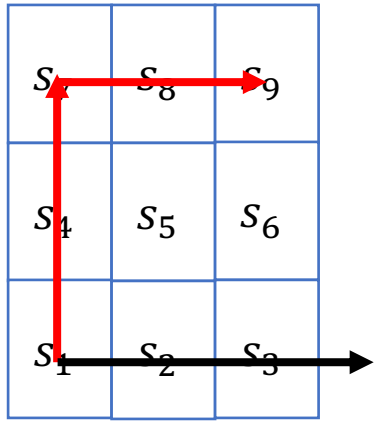
$$0 + \gamma \times 0 + \cdots + \gamma^3 \times 1 = 0.970$$

$$0 + \gamma \times 0 + \gamma^2 \times (-1) = -0.980$$

因此估计

$$V(s_1) = \frac{1}{2}(0.970 - 0.980) = -0.005$$

如果是确定的策略，每个起点只会产生一种轨迹



根据数理统计的知识，期望可以通过样本均值来估计的，这正是蒙特卡洛方法（Monte-Carlo method）的核心思想。即大数定理指出：对于独立同分布（independent and identically distributed, i.i.d)的样本数据，当样本足够大的时候，样本平均值向期望值收敛。

强化学习中的策略评估：时序差分 - 估计$V^{\pi}(s)$

- **时序差分法Temporal-difference (TD)**

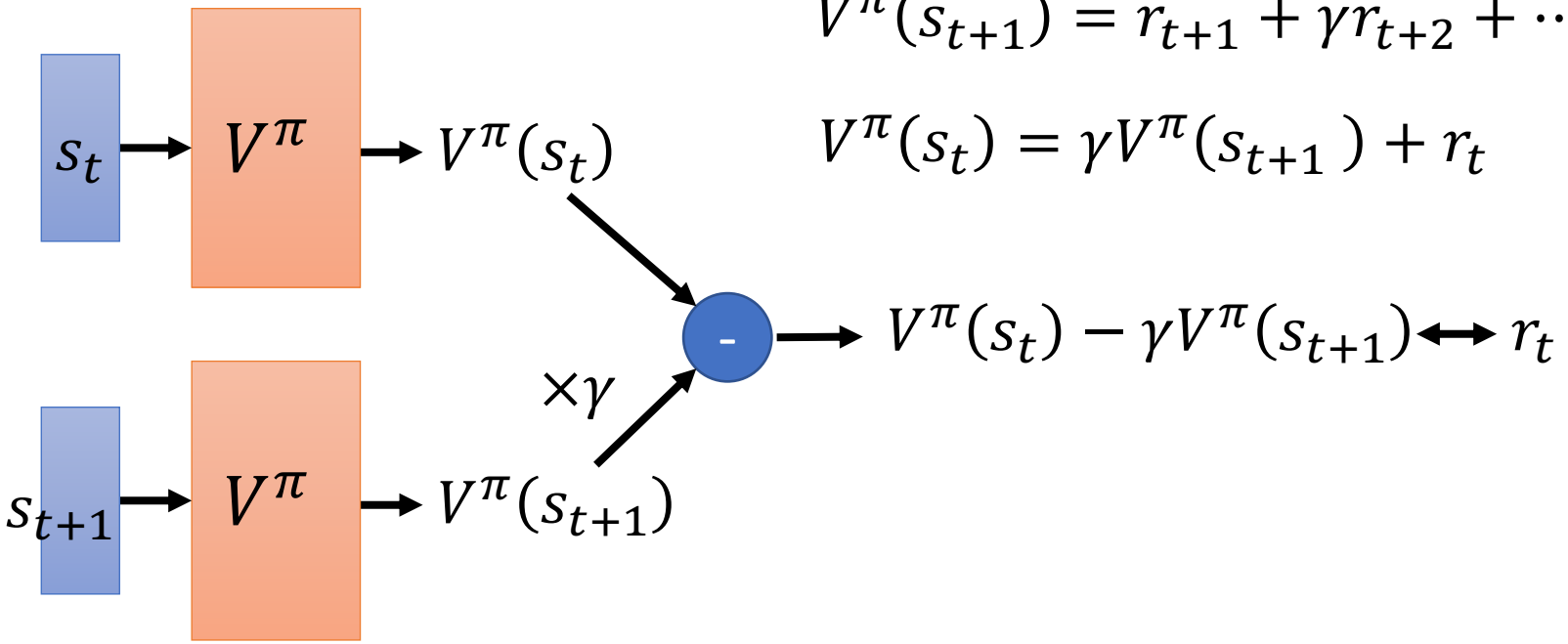
$\cdots s_t, a_t, r_t, s_{t+1} \cdots$

(ignore the expectation here)

$$V^{\pi}(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} \cdots$$

$$V^{\pi}(s_{t+1}) = r_{t+1} + \gamma r_{t+2} + \cdots$$

$$V^{\pi}(s_t) = \gamma V^{\pi}(s_{t+1}) + r_t$$



北京航空航天大學 软件学院

强化学习中的策略评估：时序差分

基于时序差分（Temporal Difference）的价值函数更新

初始化V_π函数
循环
　　初始化s为初始状态
　　循环
　　　　$a \sim \pi(s,\cdot)$
　　　　执行动作a，观察奖励R和下一个状态s'
　　　　更新$V_\pi(s) \leftarrow V_\pi(s) + \alpha[R(s,a,s') + \gamma V_\pi(s') - V_\pi(s)]$
　　　　$s \leftarrow s'$
　　直到s是终止状态
直到V_π收敛

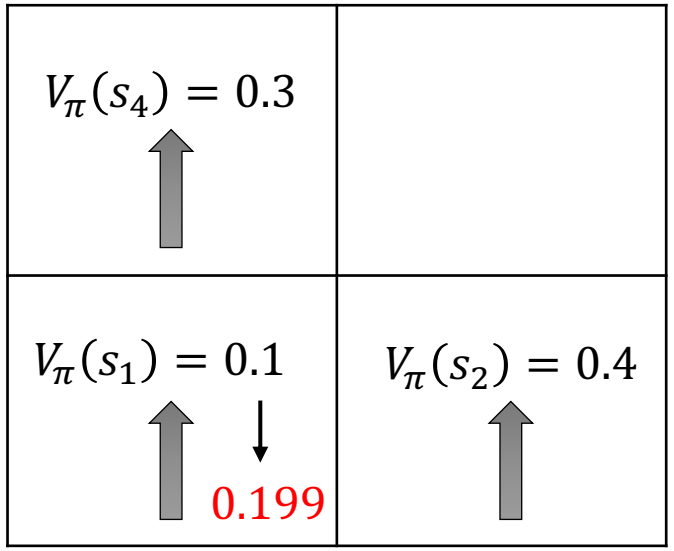
- 根据贝尔曼方程$V_\pi(s) = \mathbb{E}_{a\sim\pi(s,\cdot),s'\sim Pr(\cdot|s,a)}[R(s,a,s') + \gamma V_\pi(s')]$
- 利用蒙特卡洛采样的思想，通过采样a和s'来估计期望
- $R(s,a,s') + \gamma V_\pi(s')$是对$V_\pi(s)$的一个估计值
- 部分更新$V_\pi(s)$的值：$V_\pi(s) \leftarrow (1-\alpha)V_\pi(s) + \alpha[R(s,a,s') + \gamma V_\pi(s')]$

强化学习中的策略评估：时序差分

$$R(S_t, a_t, S_{t+1}) = \begin{cases} 1, \text{如果} S_{t+1} = s_9 \\ -1, \text{如果} S_{t+1} = s_d \\ 0, \text{其他情况} \end{cases}$$

基于时序差分（Temporal Difference）的价值函数更新

初始化V_π函数
循环
　　初始化s为初始状态
　　循环
　　　　$a \sim \pi(s,\cdot)$
　　　　执行动作a，观察奖励R和下一个状态s'
　　　　更新$V_\pi(s) \leftarrow V_\pi(s) + \alpha[R(s,a,s') + \gamma V_\pi(s') - V_\pi(s)]$
　　　　$s \leftarrow s'$
　　直到s是终止状态
直到V_π收敛



假设$\alpha = 0.5$，更新$V_\pi(s_1)$的值：
从$\pi(s_1,\cdot)$中采样得到动作$a =$ 上
从$Pr(\cdot|s_1, \text{上})$中采样得到下一步状态$s' = s_4$

$$V_\pi(s_1) \leftarrow V_\pi(s_1) + \alpha[R(s_1, \text{上}, s_4) + \gamma V_\pi(s_4) - V_\pi(s_1)]$$
$$= 0.1 + 0.5 \times [0 + 0.99 \times 0.3 - 0.1] = 0.199$$

在对片段进行采样的同时，不断以上述方法更新当前状态的价值函数，不断迭代直到价值函数收敛为止。

强化学习中的策略评估：Q-learning

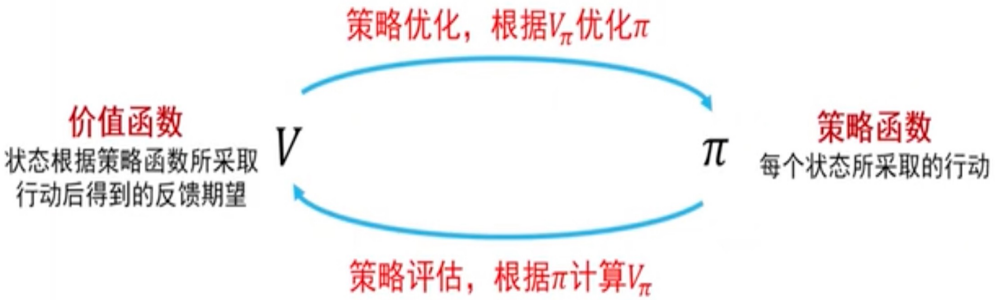


图7.7 基于通用策略迭代来学习最优策略的模式

Q-learning: 策略优化和策略评估相结合

初始化q_π函数
循环
　　初始化s为初始状态
　　循环
　　~~$a \sim \pi(s,\cdot)$~~ $a = \text{argmax}_{a'} q_\pi(s,a')$
　　执行动作a，观察奖励R和下一个状态s'
　　~~更新$V_\pi(s) \leftarrow V_\pi(s) + \alpha[R + \gamma V_\pi(s') - V_\pi(s)]$~~
　　更新$q_\pi(s,a) \leftarrow q_\pi(s,a) + \alpha\left[R + \gamma \max_{a'} q_\pi(s',a') - q_\pi(s,a)\right]$
　　$s \leftarrow s'$
　　直到s是终止状态
直到q_π收敛

策略优化：$\pi'(s) = \text{argmax}_a q_\pi(s,a)$

$$q_\pi(s,a) \leftarrow (1-\alpha) q_\pi(s,a) + \alpha\left[R + \gamma \max_{a'} q_\pi(s',a')\right]$$

Q学习中直接记录和更新动作-价值函数q_π而不是价值函数V_π，这是因为策略优化要求已知动作-价值函数q_π，如果算法仍然记录价值函数V_π，在不知道状态转移概率的情况下将无法求出q_π。于是，Q学习中，只有动作-价值函数（即q函数）参与计算。

强化学习中的策略评估：Q-learning

初始化q_π函数
循环
　　初始化s为初始状态
　　循环
　　$a = \text{argmax}_{a'} q_\pi(s, a')$
　　执行动作a，观察奖励R和下一个状态s'
　　更新$q_\pi(s,a) \leftarrow q_\pi(s,a) + \alpha\left[R + \gamma \max_{a'} q_\pi(s', a') - q_\pi(s,a)\right]$
　　$s \leftarrow s'$
　　直到s是终止状态
直到q_π收敛

s_d			
	s_7	s_8	s_9
	s_4	s_5	s_6
	s_1	s_2	s_3

初始化q_π函数

在右图中，$^a/_b$表示$q_\pi(s, 上) = a, q_\pi(s, 右) = b$

所有终止状态的q函数值设为$^0/_0$，其余状态可随机初始化，此处设$^{0.2}/_0$

初始化s，s的值在右图中用黑框框出

$^0/_0$	$^{0.2}/_0$	$^{0.2}/_0$	$^0/_0$
	$^{0.2}/_0$	$^{0.2}/_0$	$^{0.2}/_0$
	$^{0.2}/_0$	$^{0.2}/_0$	$^{0.2}/_0$

强化学习中的策略评估：Q-learning

初始化q_π函数
循环
　　初始化s为初始状态
　　循环
　　$a = \text{argmax}_{a'} q_\pi(s, a')$
　　执行动作a，观察奖励R和下一个状态s'
　　更新$q_\pi(s,a) \leftarrow q_\pi(s,a) + \alpha\left[R + \gamma \max_{a'} q_\pi(s',a') - q_\pi(s,a)\right]$
　　$s \leftarrow s'$
　　直到s是终止状态
直到q_π收敛

s_d	s_7	s_8	s_9
	s_4	s_5	s_6
	s_1	s_2	s_3

$a = \text{argmax}_{a'} q_\pi(s_1, a') = $ 上

$R = 0$，$s' = s_4$

$q_\pi(s_1, \text{上}) \leftarrow 0.2 + 0.5 \times [0 + 0.99 \times \max\{0, 0.2\} - 0.2] = 0.199$

$s \leftarrow s_4$

$0.2/0$	$0.2/0$	$0/0$
$0.2/0$	$0.2/0$	$0.2/0$
$0.199/0$	$0.2/0$	$0.2/0$

强化学习中的策略评估：Q-learning

初始化q_π函数
循环
　　初始化s为初始状态
　　循环
　　$a = \text{argmax}_{a'} q_\pi(s, a')$
　　执行动作a，观察奖励R和下一个状态s'
　　更新$q_\pi(s,a) \leftarrow q_\pi(s,a) + \alpha\left[R + \gamma \max_{a'} q_\pi(s',a') - q_\pi(s,a)\right]$
　　$s \leftarrow s'$
　　直到s是终止状态
直到q_π收敛

s_d		
s_7	s_8	s_9
s_4	s_5	s_6
s_1	s_2	s_3

$a = \text{argmax}_{a'} q_\pi(s_4, a') = $ 上

$R = 0$，$s' = s_7$

$q_\pi(s_4, \text{上}) \leftarrow 0.2 + 0.5 \times [0 + 0.99 \times \max\{0, 0.2\} - 0.2] = 0.199$

$s \leftarrow s_7$

	$0.2/0$	$0.2/0$	$0/0$
$0/0$	$0.199/0$	$0.2/0$	$0.2/0$
	$0.199/0$	$0.2/0$	$0.2/0$

强化学习中的策略评估：Q-learning

初始化q_π函数
循环
　　初始化s为初始状态
　　循环
　　$a = \text{argmax}_{a'} q_\pi(s, a')$
　　执行动作a，观察奖励R和下一个状态s'
　　更新$q_\pi(s,a) \leftarrow q_\pi(s,a) + \alpha\left[R + \gamma \max_{a'} q_\pi(s', a') - q_\pi(s,a)\right]$
　　$s \leftarrow s'$
　　直到s是终止状态
直到q_π收敛

s_7	s_8	s_9
s_4	s_5	s_6
s_1	s_2	s_3

s_d

$0/0$	$-0.4/0$	$0.2/0$	$0/0$
	$0.199/0$	$0.2/0$	$0.2/0$
	$0.199/0$	$0.2/0$	$0.2/0$

$a = \text{argmax}_{a'} q_\pi(s_7, a') =$ 上
$R = -1$，$s' = s_d$
$q_\pi(s_7, \text{上}) \leftarrow 0.2 + 0.5 \times [-1 + 0.99 \times \max\{0,0\} - 0.2] = -0.4$
$s \leftarrow s_d$

因为s_d是终止状态，因此一个片段（episode）结束

强化学习中的策略评估：Q-learning

s_d	s_7	s_8	s_9
	s_4	s_5	s_6
	s_1	s_2	s_3

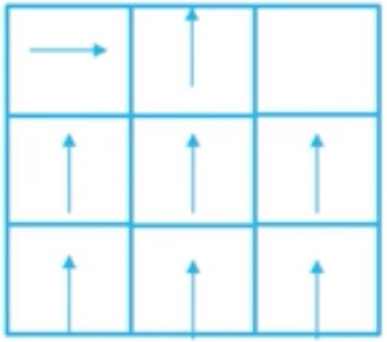
q函数

$-0.4/0$	$0.2/0$	$0/0$
$0.199/0$	$0.2/0$	$0.2/0$
$0.199/0$	$0.2/0$	$0.2/0$

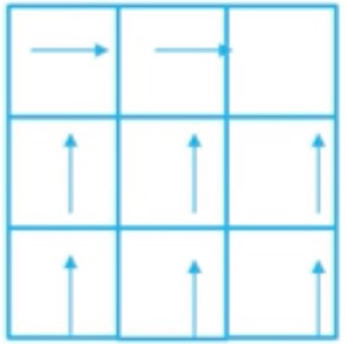
$-0.4/0.099$	$-0.4/0$	$0/0$
$0.100/0$	$0.2/0$	$0.2/0$
$0.198/0$	$0.2/0$	$0.2/0$

$-0.4/0.050$	$-0.4/0.5$	$0/0$
$0.099/0$	$0.2/0$	$0.2/0$
$0.148/0$	$0.2/0$	$0.2/0$

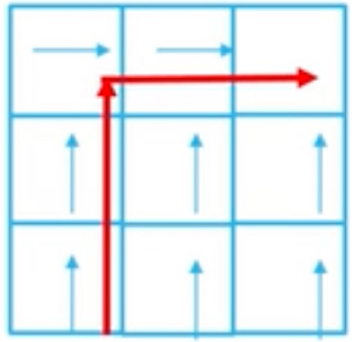
策略



第一个片段后



第二个片段后



第三个片段后

策略学习中探索（exploration）与利用（exploitation）的平衡

ϵ贪心（ϵ-greedy）策略

$$\epsilon\text{-}greedy_\pi(s) = \begin{cases} \text{argmax}_a q_\pi(s,a), & \text{以} 1-\epsilon \text{的概率} \\ \text{随机的} a \in A \ , & \text{以} \epsilon \text{的概率} \end{cases}$$

初始化q_π函数
循环
　初始化s
　循环

用ϵ贪心（ϵ-greedy）策略
代替$a = \text{argmax}_{a'} q_\pi(s,a')$

$a = \text{argmax}_{a'} q_\pi(s,a')$

执行动作a，观察奖励R和下一个状态s'

更新$q_\pi(s,a) \leftarrow q_\pi(s,a) + \alpha\left[R + \gamma \max_{a'} q_\pi(s',a') - q_\pi(s,a)\right]$

$s \leftarrow s'$

　直到s是终止状态
直到q_π收敛

- 在初步的学习中，Q值是不准确的，如果在这个时候都按照Q值来选择，那么会造成错误。
- 学习一段时间后，机器人的路线会相对固定，则机器人无法对环境进行有效的探索。

因此，在Q学习中引入探索（exploration）与利用（exploitation）机制。这一机制用用ϵ贪心（ϵ-greedy）策略来代替$a = argmax_{a'} q_\pi(s,a')$。用$\epsilon$贪心（$\epsilon$-greedy）策略定义如下：在状态s, 以$1-\epsilon$的概率来选择带来最大回报的动作，或者以$\epsilon$的概率来随机选择一个动作。

强化学习中的策略评估：使用ϵ贪心策略的Q学习

初始化q_π函数
循环
　　初始化s为初始状态
　　循环
　　采样$a \sim \epsilon - greedy_\pi(s)$
　　执行动作a，观察奖励R和下一个状态s'
　　更新$q_\pi(s,a) \leftarrow q_\pi(s,a) + \alpha\left[R + \gamma \max_{a'} q_\pi(s',a') - q_\pi(s,a)\right]$
　　$s \leftarrow s'$
　　直到s是终止状态
直到q_π收敛

采样策略与更新策略不同

- 将动作采样从“确定地选取最优动作”改为“按照ϵ贪心策略选取动作”
- 更新时仍保持用max操作选取最佳策略。像这样更新时的目标策略与采样策略不同的方法，叫做离策略（off-policy）方法

北京航空航天大学 软件学院

用神经网络拟合（动作）价值函数：Deep Q-learning

- 状态数量太多时，有些状态可能始终无法采样到，因此对这些状态的q函数进行估计是很困难的
- 状态数量无限时，不可能用一张动作-状态Q值表（数组）来记录q函数的值

使用ϵ贪心策略的Q学习

初始化q_π函数
循环
　　初始化s为初始状态
　　循环
　　采样$a \sim \epsilon\text{-}greedy_\pi(s)$
　　执行动作a，观察奖励R和下一个状态s'
　　更新$q_\pi(s,a) \leftarrow q_\pi(s,a) + \alpha\left[R + \gamma \max_{a'} q_\pi(s',a') - q_\pi(s,a)\right]$
　　$s \leftarrow s'$
　　直到s是终止状态
直到q_π收敛

思路：将q函数参数化（parametrize），用一个非线性回归模型来拟合q函数，例如（深度）神经网络

- 能够用有限的参数刻画无限的状态
- 由于回归函数的连续性，没有探索过的状态也可通过周围的状态来估计

用神经网络拟合（动作）价值函数：Deep Q-learning

用深度神经网络拟合q函数

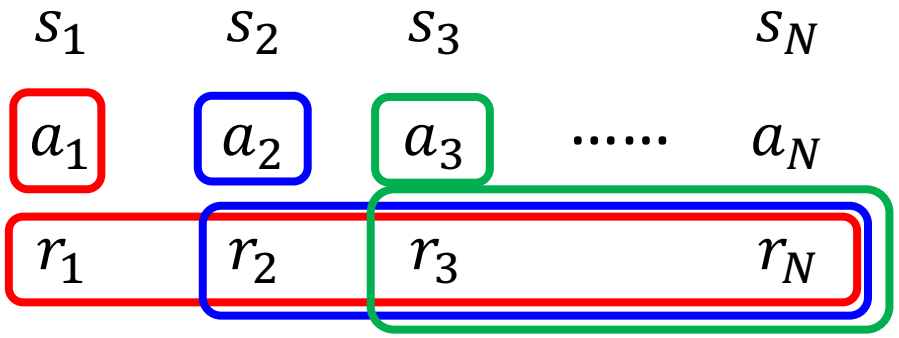
初始化$\boldsymbol{q_\pi}$函数的参数$\boldsymbol{\theta}$
循环
　　初始化s为初始状态
　　循环
　　采样$\boldsymbol{a \sim \epsilon\text{-}greedy_\pi(s;\theta)}$
　　执行动作$\boldsymbol{a}$，观察奖励$\boldsymbol{R}$和下一个状态$\boldsymbol{s'}$
　　损失函数$\boldsymbol{L(\theta) = \frac{1}{2}\left[R + \gamma \max_{a'} q_\pi(s', a'; \theta) - q_\pi(s, a; \theta)\right]^2}$
　　根据梯度$\boldsymbol{\partial L(\theta)/\partial\theta}$更新参数$\boldsymbol{\theta}$
　　$\boldsymbol{s \leftarrow s'}$
　　直到$\boldsymbol{s}$是终止状态
直到$\boldsymbol{q_\pi}$收敛

- 损失函数刻画了q的估计值$R + \gamma \max_{a'} q_\pi(s', a'; \theta)$与当前值的平方误差
- 利用梯度下降法优化参数θ
- 如果用深度神经网络来拟合q函数，则算法称为深度Q学习或者深度强化学习

北京航空航天大學 软件学院

提纲

回顾：基于策略的强化学习（版本 3）



奖励的好坏是“相对的”

如果所有的$r_n \geq 10$,则$r_n = 10$是负奖励...

减去一个基线值b ???

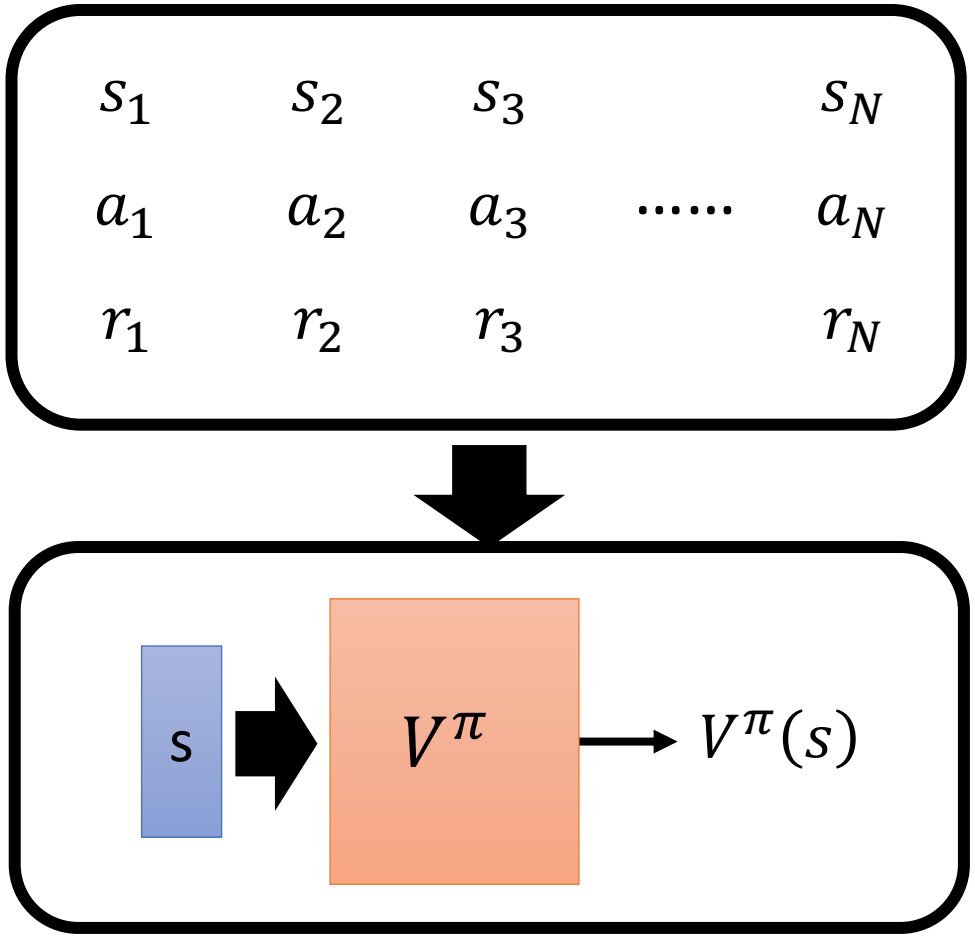
使得 G_t' 具有正负奖励值

Training Data

$$\{s_1, a_1\} \quad A_1 = G_1' - b$$
$$\{s_2, a_2\} \quad A_2 = G_2' - b$$
$$\{s_3, a_3\} \quad A_3 = G_3' - b$$
$$\vdots \qquad \vdots$$
$$\{s_N, a_N\} \quad A_N = G_N' - b$$

$$G_t' = \sum_{n=t}^{N} \gamma^{n-t} r_n$$

Actor-Critic方法（版本 3.5）



Training Data

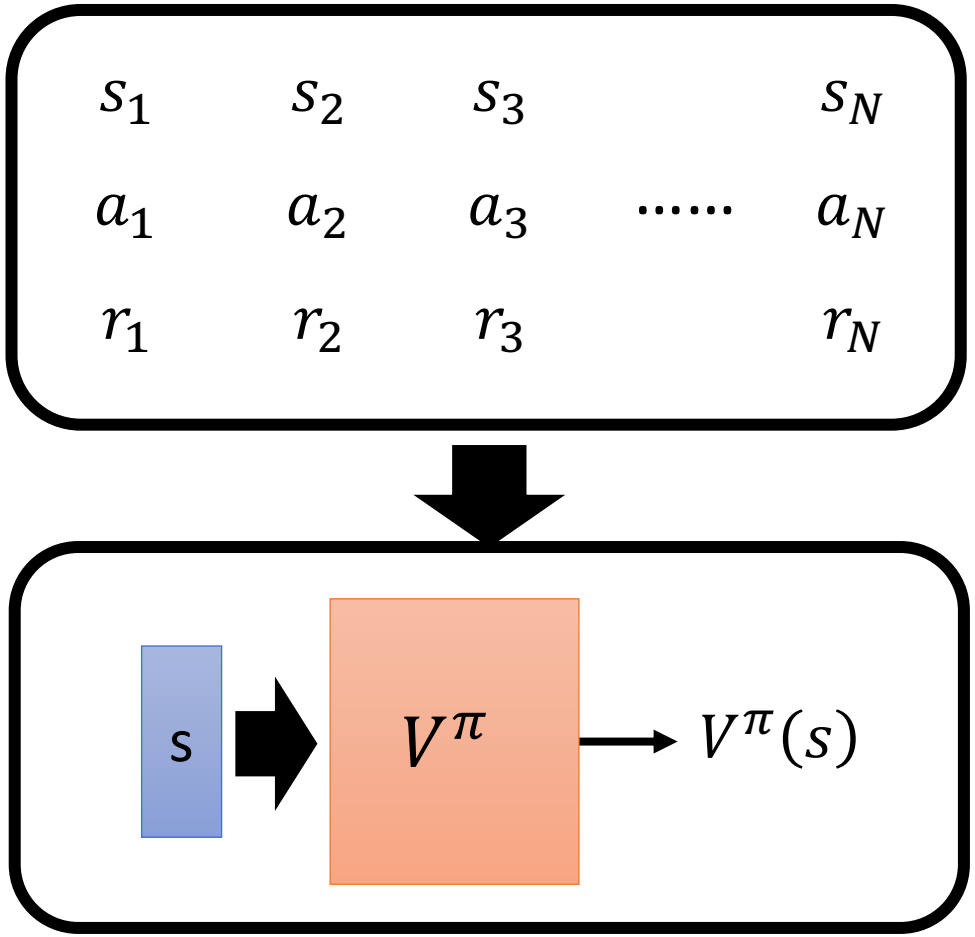
$\{s_1, a_1\}$ $\mathrm{A}_1 = G'_1 - b$

$\{s_2, a_2\}$ $\mathrm{A}_2 = G'_2 - b$

$\{s_3, a_3\}$ $\mathrm{A}_3 = G'_3 - b$

$\{s_N, a_N\}$ $\mathrm{A}_N = G'_N - b$

Actor-Critic方法（版本 3.5）



Training Data

$\{s_1, a_1\}$ $\mathrm{A}_1 = G'_1 - V^\pi(s_1)$

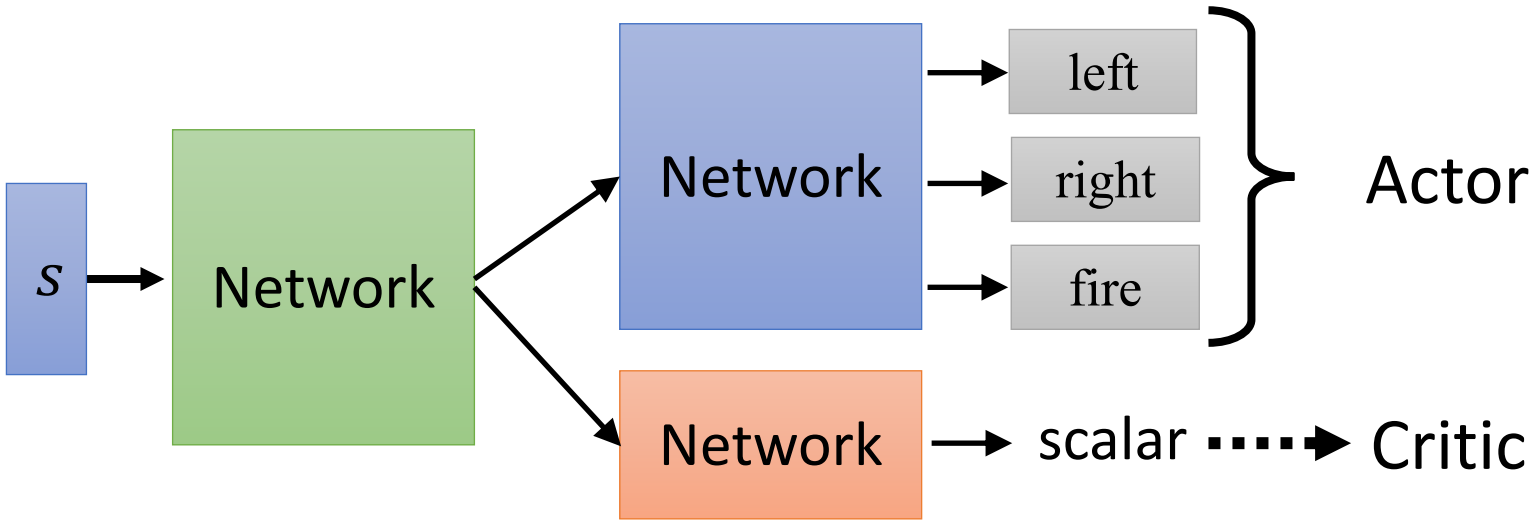
$\{s_2, a_2\}$ $\mathrm{A}_2 = G'_2 - V^\pi(s_2)$

$\{s_3, a_3\}$ $\mathrm{A}_3 = G'_3 - V^\pi(s_3)$

$\{s_N, a_N\}$ $\mathrm{A}_N = G'_N - V^\pi(s_N)$

Tip of Actor-Critic

- 策略模型(Actor)和评判模型(Critic)参数可以共享



北京航空航天大学 软件学院

提纲

Reward Shaping

稀疏奖励Sparse Reward

$$A_t = r_t + V^{\pi}(s_{t+1}) - V^{\pi}(s_t)$$

s_1	s_2	s_3		s_N
a_1	a_2	a_3	……	a_N
r_1	r_2	r_3		r_N

Training Data

$\{s_1, a_1\}$ A_1

$\{s_2, a_2\}$ A_2

$\{s_3, a_3\}$ A_3

⋮ ⋮

$\{s_N, a_N\}$ A_N

如何多数情况 $r_t = 0$ ➡ 难以判断某个动作的好坏

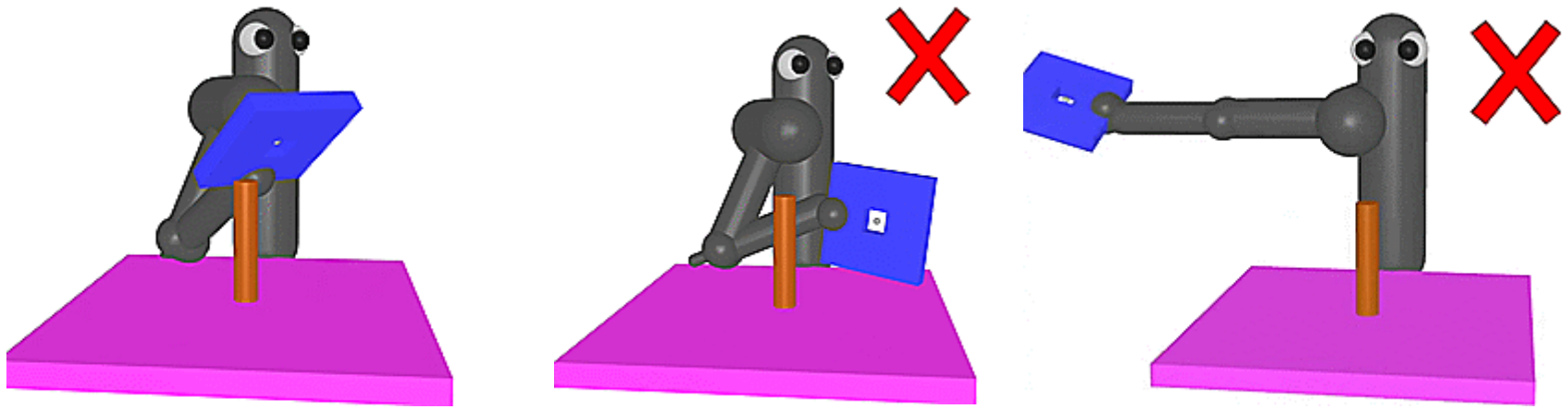
e.g., 机械手臂栓螺丝

定义额外奖励来指导智能体的学习 ➡ ***reward shaping***

Reward Shaping

VizDoom https://openreview.net/forum?id=Hk3mPK5gg&noteId=Hk3mPK5gg

Parameters	Description	FlatMap | CIGTrack1
living	Penalize agent who just lives	-0.008 / action
health_loss	Penalize health decrement	-0.05 / unit
ammo_loss	Penalize ammunition decrement	-0.04 / unit
health_pickup	Reward for medkit pickup	0.04 / unit
ammo_pickup	Reward for ammunition pickup	0.15 / unit
dist_penalty	Penalize the agent when it stays	-0.03 / action
dist_reward	Reward the agent when it moves	9e-5 / unit distance



https://bair.berkeley.edu/blog/2017/12/20/reverse-curriculum/

No Reward:
Learning from Demonstration

动机

- 有些任务难以定义奖励。
- 手工设计奖励可能会导致不可控行为

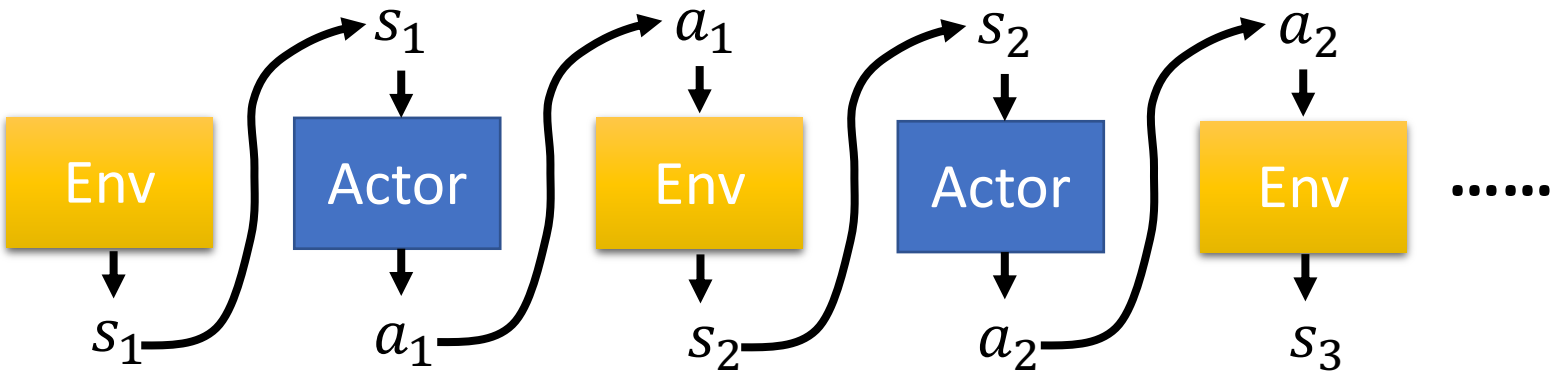
Three Laws of Robotics:

1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.
2. A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.
3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Laws.

?????

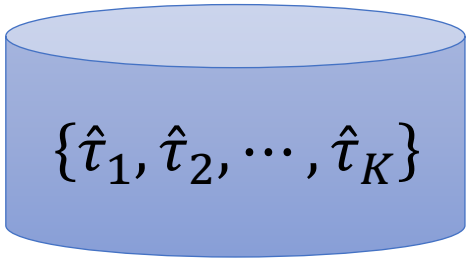
restraining individual human behavior and sacrificing some humans will ensure humanity's survival

模仿学习 Imitation Learning



智能体可以与环境交互，但是奖励函数未知

专家示范



$\hat{\tau}$ 为专家轨迹

自动驾驶：记录司机驾驶行为

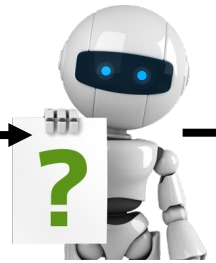
机器人：握住机器人手臂抓取

监督学习?

- 以自动驾驶为例

$$\hat{\tau} = \{s_1, \hat{a}_1, s_2, \hat{a}_2, \cdots\}$$

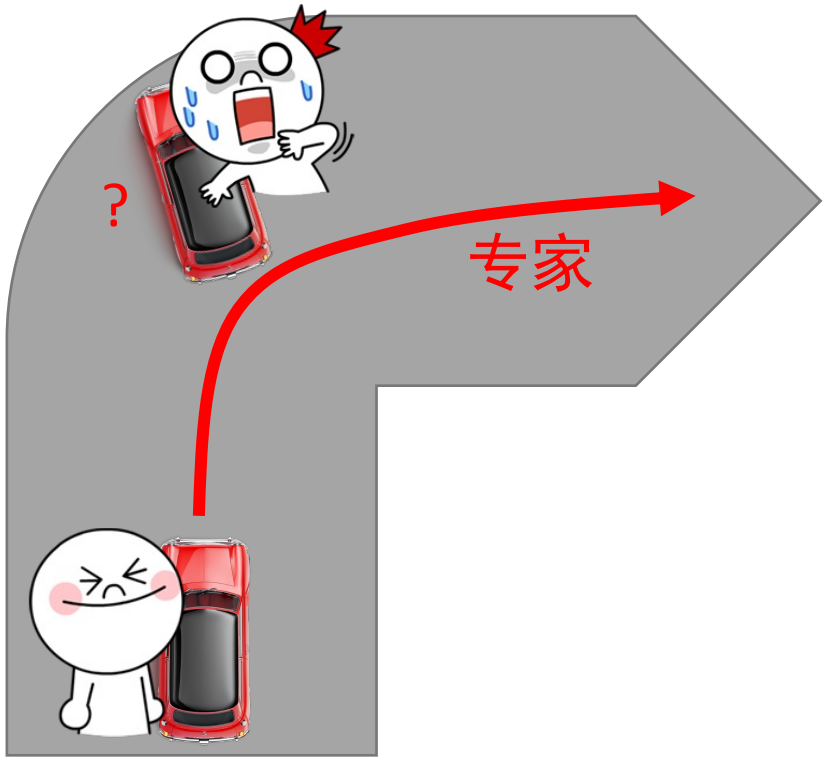
forward



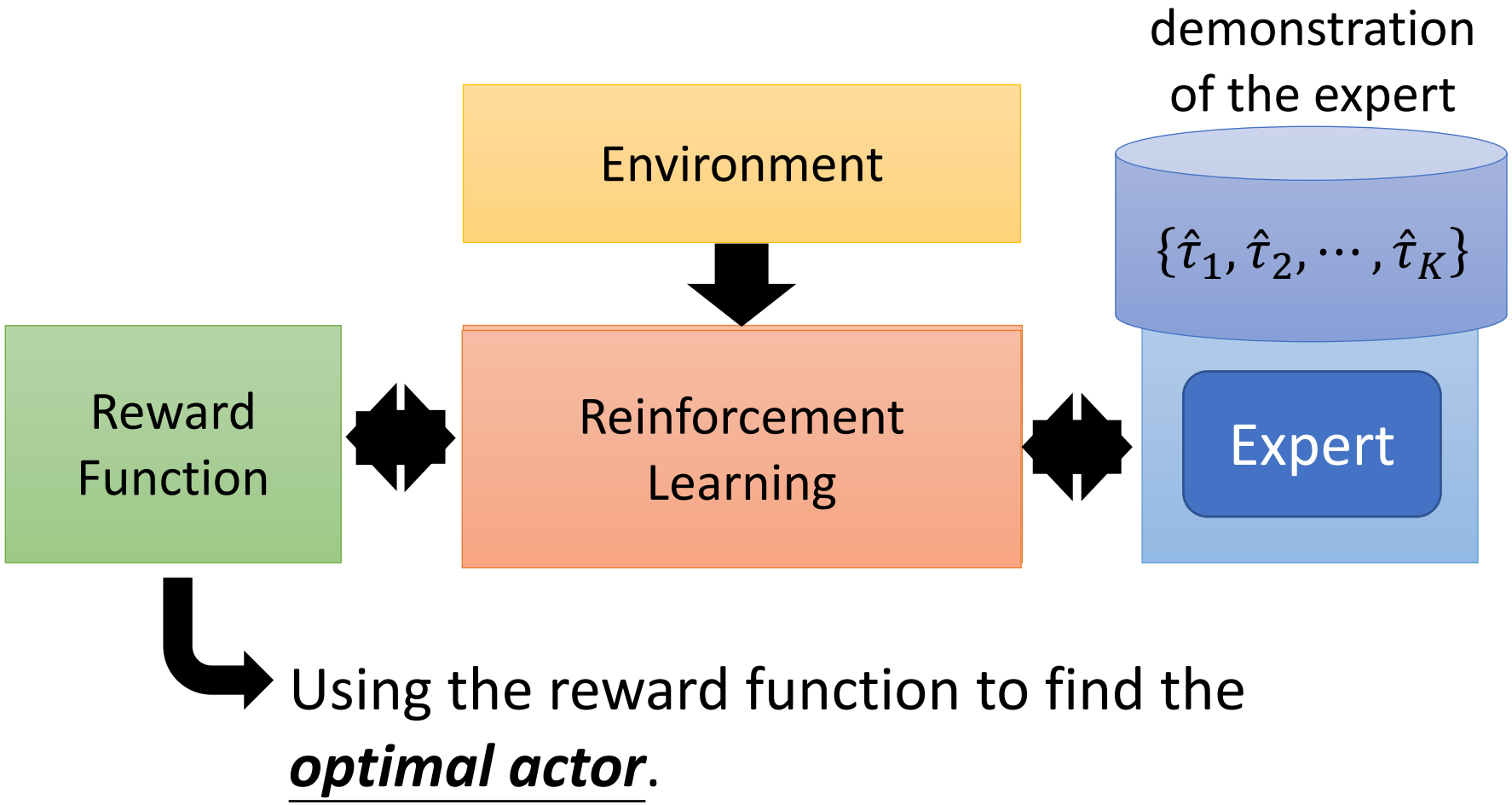
$s_i \rightarrow a_i \dashleftarrow\dashrightarrow \hat{a}_i$

问题：只能采样有限数据

Yes, also known as *Behavior Cloning*



逆强化学习 Inverse Reinforcement Learning



更多阅读

- 书籍: Reinforcement Learning: An Introduction
  - http://incompleteideas.net/sutton/book/the-book.html