

# 第16章 强化学习

2019年7月8日 星期一 上午9:44

## 16.1 任务与奖赏

强化学习任务通常使用马尔可夫决策过程 (MDP) 来描述：

机器处于环境E中，状态空间为X（状态指的是机器感知到的环境的描述），机器能采取的动作构成了动作空间A，

当某个总做a作用在当前状态x上，则潜在转移函数P使得以某个概率转移到另一个状态

对应四元组<X,A,P,R>,其中P指明了状态转移概率，R指明了奖赏

策略 (policy)  $\pi$  .

确定性策略:  $\pi: X \mapsto A$

概率表示:  $\pi: X \times A \mapsto R$   $\pi(x,a)$  表示状态 x 下选择 A 的概率

策略的优劣取决于长期执行策略后的奖赏

常用的长期累积奖赏

T步累计奖赏:  $E \left[ \frac{1}{T} \sum_{t=1}^T r_t \right]$

折扣累计奖赏:  $E \left[ \sum_{t=0}^{\infty} \gamma^t r_{t+1} \right]$

## 16.2 K-摇臂赌博机

### 16.2.1 探索与利用

先考虑最大化单步奖赏：需考虑每个动作带来的奖赏，并执行奖赏最大的动作

但问题在于，一般情况下，一个动作的奖赏是来自于一个概率分布，仅通过以此尝试并不能确切的获得平均奖赏值

最大化单步奖赏对应问题：K-摇臂赌博机（有多个摇杆，投入一个硬币后按下一个摇杆有一定概率突出硬币）

仅探索：将所有尝试机会平均分配给每一个摇臂，取平均值（能够很好的估计每一个摇杆的奖励，但失去了很多选取最优的机会）

仅利用：每次将机会给当前最优的摇臂，多个最优时随机选取一个

探索-利用窘境 (Exploration-Exploitation dilemma) : 由于探索与利用的次数有限，加强一方必然削弱另一方

### 16.2.2 $\epsilon$ -贪心

在每次尝试时，以 $\epsilon$ 的概率进行探索，即以均匀概率随机选择一个动作；以 $1-\epsilon$ 的概率进行利用，即选择当前最优的动作。 $\epsilon$ -贪心法只需记录每个动作的当前平均奖赏值与被选中的次数，便可以增量式更新。

若摇臂奖赏的不确定性较大，如概率分布较宽时则进行更多的探索，需要较大的 $\epsilon$ 值

若摇臂不确定性比较小，如概率分布比较集中时，则 $\epsilon$ 取较小的常数

通常取 $\epsilon$ 较小，如0.1或0.01

如果尝试次数非常大，则一段时间后，摇臂的奖赏可以很好的近似出来，不再需要探索，可以让 $\epsilon$ 随探索次数增加而逐渐减少

### 16.2.3 Softmax

基于已知的摇臂平均奖赏来对探索与利用进行折中，若摇臂的平均奖赏高于其他摇臂，则他们被选取的概率也明显更高

摇臂概率分配的基于Boltzmann分布

$$P(k) = \frac{e^{\frac{\alpha_k(k)}{T}}}{\sum_{i=1}^k e^{\frac{\alpha_i(i)}{T}}} \quad \alpha(i): \text{当前摇臂的平均奖赏.} \\ T: \text{温度. 越小则平均奖赏越高的摇臂被选概率越高}$$

## 16.3 有模型学习

考虑多步强化学习任务，假定任务对应的马尔可夫决策过程四元组已知（模型已知）

### 16.3.1 策略评估

考虑多步强化学习任务，假定任务对应的马尔可夫决策过程四元组已知（模型已知）

### 16.3.1 策略评估

定义  $V^\pi(x)$ : 从状态  $x$  出发，使用策略  $\pi$  所带来的奖赏。（状态值函数）

$Q^\pi(x, a)$ : 从状态  $x$  出发，执行动作  $a$  后再使用策略  $\pi$  所带来的累积奖赏  
(状态-动作值函数)

$$\begin{cases} V_T^\pi(x) = E_\pi \left[ \frac{1}{T} \sum_{t=1}^T r_t \mid x_0 = x \right] & T\text{步累积奖赏.} \\ V_r^\pi(x) = E_\pi \left[ \sum_{t=0}^{+\infty} \gamma^t r_{t+1} \mid x_0 = x \right] & \gamma\text{折扣累积奖赏} \end{cases}$$

值函数有很简单的递归形式：

$$\begin{aligned} V_T^\pi(x) &= \sum_{a \in A} \pi(x, a) \sum_{x' \in X} P_{x \rightarrow x'}^a \left( \frac{1}{T} R_{x \rightarrow x'}^a + \frac{T-1}{T} V_{T-1}^\pi(x') \right) \\ V_r^\pi(x) &= \sum_{a \in A} \pi(x, a) \sum_{x' \in X} P_{x \rightarrow x'}^a \left( R_{x \rightarrow x'}^a + \gamma V_r^\pi(x') \right) \end{aligned}$$

本质就是动态规划。从  $V_0$  出发，多次迭代即可。

通常设置一个停止阈值，当  $\max |V(x) - V'(x)| < 0$  时停止。

同理 可计算

$$\begin{cases} Q_T^\pi(x, a) = \sum_{x' \in X} P_{x \rightarrow x'}^a \left( \frac{1}{T} R_{x \rightarrow x'}^a + \frac{T-1}{T} V_{T-1}^\pi(x') \right) \\ Q_r^\pi(x, a) = \sum_{x' \in X} P_{x \rightarrow x'}^a \left( R_{x \rightarrow x'}^a + \gamma V_r^\pi(x') \right) \end{cases}$$

### 16.3.2 策略改进

理想策略：最大化累积奖赏： $\pi^* = \arg \max \sum_{x \in X} V^\pi(x)$

最优策略对应值函数  $V^*$

$$V^*(x) = \max_{a \in A} Q^{\pi^*}(x, a)$$

由于最优化函数的累积奖赏值已达最大，因此可对前面的 Bellman 等式(16.7)和(16.8)做一个改动，即将对动作的求和改为取最优：

$$\begin{cases} V_T^*(x) = \max_{a \in A} \sum_{x' \in X} P_{x \rightarrow x'}^a \left( \frac{1}{T} R_{x \rightarrow x'}^a + \frac{T-1}{T} V_{T-1}^*(x') \right); \\ V_\gamma^*(x) = \max_{a \in A} \sum_{x' \in X} P_{x \rightarrow x'}^a \left( R_{x \rightarrow x'}^a + \gamma V_\gamma^*(x') \right). \end{cases} \quad (16.13)$$

换言之，

$$V^*(x) = \max_{a \in A} Q^{\pi^*}(x, a). \quad (16.14)$$

代入式(16.10)可得最优状态-动作值函数

$$\begin{cases} Q_T^*(x, a) = \sum_{x' \in X} P_{x \rightarrow x'}^a \left( \frac{1}{T} R_{x \rightarrow x'}^a + \frac{T-1}{T} \max_{a' \in A} Q_{T-1}^*(x', a') \right); \\ Q_\gamma^*(x, a) = \sum_{x' \in X} P_{x \rightarrow x'}^a \left( R_{x \rightarrow x'}^a + \gamma \max_{a' \in A} Q_\gamma^*(x', a') \right). \end{cases} \quad (16.15)$$

上述关于最优化函数的等式，称为最优 Bellman 等式，其唯一解是最优化函数。

**非优策略改进方式：将策略选择动作改变为当前最优的动作**

$$Q^\pi(x, \pi'(x)) \geq V^\pi(x)$$

### 16.3.3 策略迭代与值迭代

获得最优解：从一个初始策略出发（通常为随机策略），先进行策略评估，然后改进策略，评估改进策略，再改变策略...不断进行策略改变与评估，直至策略收敛，这称为策略迭代 (policy iteration)

但是由于这样比较耗时，而策略改进与值函数的改进是一致的，因此就可以将策略改进视为值函数的改善

$$\begin{cases} V_T(x) = \max_{a \in A} \sum_{x' \in X} P_{x \rightarrow x'}^a \left( \frac{1}{T} R_{x \rightarrow x'}^a + \frac{T-1}{T} V_{T-1}(x') \right); \\ V_\gamma(x) = \max_{a \in A} \sum_{x' \in X} P_{x \rightarrow x'}^a \left( R_{x \rightarrow x'}^a + \gamma V_\gamma(x') \right). \end{cases} \quad (16.18)$$

于是可得到值迭代(value iteration)算法，如图 16.9 所示。

---

输入：MDP 四元组  $E = \langle X, A, P, R \rangle$ ;  
累积奖赏参数  $T$ ;  
收敛阈值  $\theta$ .

过程：

- 1:  $\forall x \in X : V(x) = 0;$
- 2: **for**  $t = 1, 2, \dots$  **do**
- 3:    $\forall x \in X : V'(x) = \max_{a \in A} \sum_{x' \in X} P_{x \rightarrow x'}^a \left( \frac{1}{t} R_{x \rightarrow x'}^a + \frac{t-1}{t} V(x') \right);$
- 4:   **if**  $\max_{x \in X} |V(x) - V'(x)| < \theta$  **then**
- 5:     **break**
- 6:   **else**
- 7:      $V = V'$
- 8:   **end if**
- 9: **end for**

输出：策略  $\pi(x) = \arg \max_{a \in A} Q(x, a)$

---

图 16.9 基于  $T$  步累积奖赏的值迭代算法

若采用  $\gamma$  折扣累积奖赏，只需将图 16.9 算法中第 3 行替换为

$$\forall x \in X : V'(x) = \max_{a \in A} \sum_{x' \in X} P_{x \rightarrow x'}^a \left( R_{x \rightarrow x'}^a + \gamma V(x') \right). \quad (16.19)$$

### 16.4 免模型学习

若算法不依赖环境建模，则称为免模型学习 (model-free learning)

### 16.4.1 蒙特卡罗强化学习

在模型未知的情况下，从起始状态出发，使用某种策略进行采样，执行该策略T步并获得轨迹，然后对其中出现的每一对状态-动作的奖赏进行累计，多次采样获得多条轨迹后，对每个状态-动作的累计奖赏进行平均，得到状态-动作函数值的估计

如果要获得较好的值函数估计，需要多条不同的轨迹，如果策略是确定的，即对于某个状态只能固定的输出一个动作，则其与K摇臂赌博机中的“仅利用”是相似的，故而我们可以借鉴探索与利用的折中办法，如采取 $\epsilon$ -贪心法

$$\pi^{\epsilon}(x) = \begin{cases} \pi(x) & \text{以概率 } 1 - \epsilon \\ A \text{ 中以均匀概率选取的动作} & \text{以概率 } \epsilon. \end{cases}$$

### 16.4.2 时序差分学习

### 16.5 值函数近似

### 16.6 模仿学习

#### 16.6.1 直接模仿学习

#### 16.6.2 逆强化学习