马尔可夫决策过程的贝尔曼方程、贝尔曼期望方程

马尔可夫奖励过程与马尔可夫决策过程

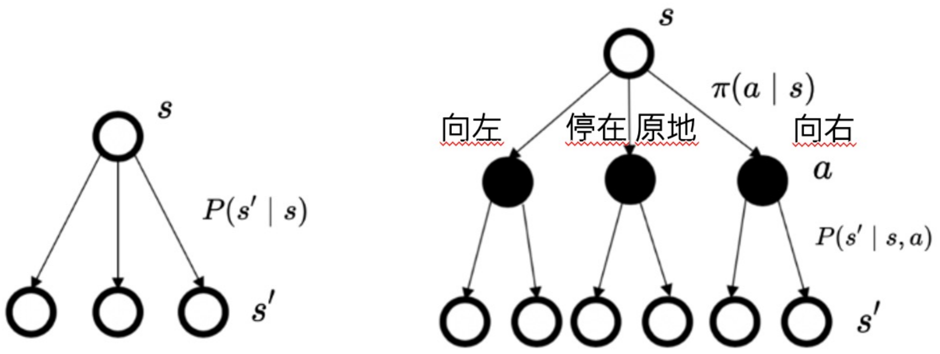


图 2.9 马尔可夫决策过程与马尔可夫过程/马尔可夫奖励过程的状态转移的对比

比马尔可夫奖励过程多了决策层

马尔可夫奖励过程MRP的预测问题：是给定一个马尔可夫奖励过程，我们要确定每个状态的价值是多少。

$$V(s)=\underbrace{R(s)}_{\text{即时奖励}}+\underbrace{\gamma \sum_{s' \in S} p\left(s' \mid s\right) V\left(s'\right)}_{\text{未来奖励的折扣总和}}$$

MDP的贝尔曼方程和MRP的是否相似？是的

马尔可夫决策过程MDP的预测问题：给定MDP与策略，我们要确定每个状态的价值是多少

MDP的预测问题：贝尔曼方程

MRP的贝尔曼方程

$$V(s)=\underbrace{R(s)}_{\text{即时奖励}}+\underbrace{\gamma\sum_{s'\in S}p\left(s'\mid s\right)V\left(s'\right)}_{\text{未来奖励的折扣总和}}$$

V价值和Q价值的相互转换

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

某状态的价值=该状态下所有可能的动作*在该状态下采取该动作的价值

MDP的贝尔曼方程

$$\begin{aligned}
Q(s,a) &= \mathbb{E}\left[G_t \mid s_t=s,a_t=a\right] \\
&= \mathbb{E}\left[r_{t+1}+\gamma r_{t+2}+\gamma^2 r_{t+3}+\ldots \mid s_t=s,a_t=a\right] \\
&= \mathbb{E}\left[r_{t+1}|s_t=s,a_t=a\right]+\gamma\mathbb{E}\left[r_{t+2}+\gamma r_{t+3}+\gamma^2 r_{t+4}+\ldots \mid s_t=s,a_t=a\right] \\
&= R(s,a)+\gamma\mathbb{E}[G_{t+1}|s_t=s,a_t=a] \\
&= R(s,a)+\gamma\mathbb{E}[V(s_{t+1})|s_t=s,a_t=a] \\
&= R(s,a)+\gamma\sum_{s'\in S}p\left(s'\mid s,a\right)V\left(s'\right)
\end{aligned}$$

贝尔曼期望方程

MRP的贝尔曼方程

$$V(s)=\underbrace{R(s)}_{\text{即时奖励}}+\underbrace{\gamma\sum_{s'\in S}p\left(s'\mid s\right)V\left(s'\right)}_{\text{未来奖励的折扣总和}}$$

两边都是V，可以迭代

MDP的贝尔曼方程

$$Q(s,a)=R(s,a)+\gamma\sum_{s'\in S}p\left(s'\mid s,a\right)V\left(s'\right)$$

左边Q右边V，不太方便

V价值和Q价值的相互转换

↓两边都是Q，可以迭代了

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

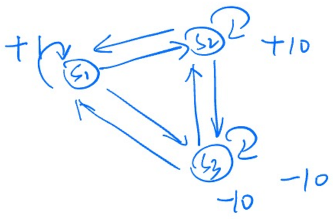
带入到上式的V中

贝尔曼期望方程（变形）

$$\begin{cases}Q_\pi(s,a)=R(s,a)+\gamma\sum_{s'\in S}p\left(s'\mid s,a\right)\sum_{a'\in A}\pi\left(a'\mid s'\right)Q_\pi\left(s',a'\right)\\[2ex] V_\pi(s)=\sum_{a\in A}\pi(a\mid s)\left(R(s,a)+\gamma\sum_{s'\in S}p\left(s'\mid s,a\right)V_\pi\left(s'\right)\right)\end{cases}$$

贝尔曼期望方程
手算

$$Q_\pi(s,a) = R(s,a) + \gamma \sum_{s'\in S} p(s' \mid s,a) \sum_{a'\in A} \pi(a' \mid s') Q_\pi(s',$$



$R=[1,10,-10]$

$\pi=(\frac{1}{3},\frac{1}{3},\frac{1}{3})$

starting from s_1

P	s_1	s_2	s_3
a_1	0.5	0.2	0.3
a_2	0.3	0.5	0.2
a_3	0.1	0.3	0.6

init

Q	s_1	s_2	s_3
a_1	0	0	0
a_2	0	0	0
a_3	0	0	0

starting from s_2

P	s_1	s_2	s_3
a_1	0.5	0.1	0.4
a_2	0.2	0.6	0.2
a_3	0.2	0.2	0.6

starting from s_3

P	s_1	s_2	s_3
a_1	0.5	0.4	0.1
a_2	0.1	0.5	0.4
a_3	0.3	0.1	0.6

Bellman: $Q(s,a)=R(s,a)+\sum_{s'\in S} p(s'|s,a)\sum_{a'\in A}\pi(a'|s')Q_\pi(s',a')$

Round 1 $Q(s_1,a_1)= 1+ p(s_1|s_1,a_1)\big(\pi(a_1|s_1)Q_\pi(s_1,a_1)+\pi(a_2|s_1)Q_\pi(s_1,a_2)+\pi(a_3|s_1)Q_\pi(s_1,a_3)\big)$
$+p(s_2|s_1,a_1)\big(\pi(a_1|s_2)Q_\pi(s_2,a_1)+\pi(a_2|s_2)Q_\pi(s_2,a_2)+\pi(a_3|s_2)Q_\pi(s_2,a_3)\big)$
$+p(s_3|s_1,a_1)V(s_3)\big(\pi(a_1|s_3)Q_\pi(s_3,a_1)+\pi(a_2|s_3)Q_\pi(s_3,a_2)+\pi(a_3|s\ )Q_\pi(s\ ,a_3)\big)$

$= 1$

$Q(s_1,a_2)=1$ $Q(s_1,a_3)=1$ $Q(s_2,a_1)=Q(s_2,a_3)=Q(s_2,a_2)=10$
$Q(s_3,a_1)=Q(s_3,a_2)=Q(s_3,a_3)=-10.$

after round 1

Q	s_1	s_2	s_3
a_1	1	10	−10
a_2	1	10	−10
a_3	1	10	−10

$V(s)=\sum_{a\in A}\pi(a|s)Q_\pi(s,a) \Rightarrow V:$

	s_1	s_2	s_3
V	1	10	−10

Round 2. $Q(s_1,a_1)= 1+ p(s_1|s_1,a_1)\big(\pi(a_1|s_1)Q_\pi(s_1,a_1)+\pi(a_2|s_1)Q_\pi(s_1,a_2)+\pi(a_3|s_1)Q_\pi(s_1,a_3)\big)$
$+p(s_2|s_1,a_1)\big(\pi(a_1|s_2)Q_\pi(s_2,a_1)+\pi(a_2|s_2)Q_\pi(s_2,a_2)+\pi(a_3|s_2)Q_\pi(s_2,a_3)\big)$
$+p(s_3|s_1,a_1)V(s_3)\big(\pi(a_1|s_3)Q_\pi(s_3,a_1)+\pi(a_2|s_3)Q_\pi(s_3,a_2)+\pi(a_3|s\ )Q_\pi(s\ ,a_3)\big)$

Round 3.4.5…n converge

a1:向1运动，a2：向2运动，a3向3运动

贝尔曼期望方程
计算机模拟

$$Q_\pi(s,a)=R(s,a)+\gamma\sum_{s'\in S}p\left(s'\mid s,a\right)\sum_{a'\in A}\pi\left(a'\mid s'\right)Q_\pi\left(s',a'\right)$$

```python
# 定义状态、动作和奖励
states = [1, 2, 3]
actions = [1, 2, 3]  # 分别代表停留、向下个状态、向上个状态
rewards = {1: 1, 2: 10, 3: -10}

# 定义策略概率
policy = {1: 1/3, 2: 1/3, 3: 1/3}

# 定义折扣系数
gamma = 0.9

# 定义状态转移概率
transition_probs = {
    1:
    {1: {1: 0.5, 2: 0.2, 3: 0.3},
    2: {1: 0.3, 2: 0.5, 3: 0.2},
    3: {1: 0.1, 2: 0.3, 3: 0.6}},
    2:
    {1: {1: 0.5, 2: 0.1, 3: 0.4},
    2: {1: 0.2, 2: 0.6, 3: 0.2},
    3: {1: 0.2, 2: 0.2, 3: 0.6}},
    3:
    {1: {1: 0.5, 2: 0.4, 3: 0.1},
    2: {1: 0.1, 2: 0.5, 3: 0.4},
    3: {1: 0.3, 2: 0.1, 3: 0.6}}
}

# 初始化Q表格
Q = {(s, a): 0 for s in states for a in actions}
```

```python
# 初始化Q表格
Q = {(s, a): 0 for s in states for a in actions}

# 迭代计算Q函数，直到收敛
tolerance = 1e-6
cnt = 0
while True:
    cnt += 1
    Q_new = {(s, a): rewards[s]+ gamma * sum((transition_probs[s][a][s1] * sum((policy[a1]*Q[s1, a1])
                                    for a1 in actions)) for s1 in states) for (s, a) in Q}
    max_diff = max(abs(Q_new[s, a] - Q[s, a]) for (s, a) in Q)
    if max_diff < tolerance:
        break
    Q = Q_new
    print(f'第{cnt}轮迭代结果',Q)
```

```
第1轮迭代结果 {(1, 1): 1.0, (1, 2): 1.0, (1, 3): 1.0, (2, 1): 10.0, (2, 2): 10.0, (2, 3): 10.0, (3, 1): -10.0, (3, 2): -10.0, (3, 3): -10.0}
第2轮迭代结果 {(1, 1): 0.55, (1, 2): 3.9699999999999998, (1, 3): -1.6099999999999999, (2, 1): 7.75, (2, 2): 13.780000000000001, (2, 3): 6.58, (3, 1): -6.85, (3, 2): -9.01,
第3轮迭代结果 {(1, 1): 0.4150000000000005, (1, 2): 3.673000000000001, (1, 3): -1.7989999999999995, (2, 1): 7.6690000000000005, (2, 2): 13.429, (2, 3): 6.445, (3, 1): -7.09
```

第118轮迭代结果 {(1, 1): -1.397660000405915, (1, 2): **1.8650584461960262**, (1, 3): -3.6100871848719347,
(状态2, 动作1): 5.854767184060104, (2, 2): **11.622631261730007**, (2, 3): 4.632339999594086,
(3, 1): **-8.902514369337954**, (3, 2): -11.114941553803973, (3, 3): -16.125232815939896}

虽然每种动作概率相同，但价值不同

提升动作2的概率从而获得更大价值

贝尔曼期望方程
计算机模拟

第118轮迭代结果 {(1, 1): -1.397660000405915, (1, 2): **1.8650584461960262**, (1, 3): -3.6100871848719347,
(状态2, 动作1): 5.854767184060104, (2, 2): **11.622631261730007**, (2, 3): 4.632339999594086,
(3, 1): **-8.902514369337954**, (3, 2): -11.114941553803973, (3, 3): -16.125232815939896}

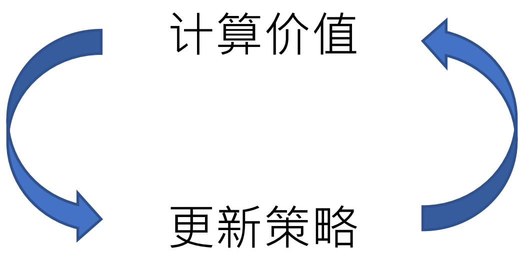
虽然每种动作概率相同，但价值不同

提升动作2的概率，产生新的策略，从而获得更大价值

计算新的策略下各状态的价值

继续更新...

逼近最优策略π※



本章小结

- 什么是马尔可夫决策过程的预测问题？
- MDP的贝尔曼方程是什么？在迭代时有什么问题？
- MDP的贝尔曼期望方程是什么？解决了问题吗？
- 计算出当前策略下的各状态价值后如何改进策略？

下一章：策略迭代与价值迭代

Credit goes to: EasyRL