Chapter 3 Finite Markov Decision Processes

马尔可夫决策过程 (Markov Decision Processes, MDPs)。

第一部分：马尔可夫决策过程 (MDP) 的世界观

1. 直觉与生活例子

想象你在玩超级马里奥（Super Mario）：

- **你 (Agent)**：控制马里奥的那个“大脑”。
- **环境 (Environment)**：游戏机、屏幕上的关卡、敌人、砖块。
- **状态 (S_t)**：当前屏幕显示的画面（马里奥的位置、敌人的位置）。
- **动作 (A_t)**：你按下的手柄按键（向右跑、跳跃）。
- **奖励 (R_{t+1})**：你采取动作后得到的反馈（吃到金币+100分，碰到敌人死掉-1000分，或者只是向前走了一步没有得分0分）。

核心循环：你看到画面（状态），按下按钮（动作），游戏引擎更新画面（新状态）并给你分数（奖励）。这个循环不断重复，这就是 MDP。

2. 数学定义与符号拆解

在数学上，我们将这个过程形式化为**有限马尔可夫决策过程 (Finite MDP)**。

- **交互序列**：

  时间步 $t = 0, 1, 2, \ldots$。交互过程产生的一串轨迹如下：

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

  注意：R_{t+1} 的下标是 $t+1$，因为它是执行了 A_t **之后**才收到的反馈。

- **环境动力学 (Dynamics) - 最核心的函数**：

  游戏引擎的内部规则是什么？在 RL 中，我们用条件概率分布 p 来描述环境的物理法则：

$$p(s', r|s, a) \doteq \mathbb{P}(S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a)$$

符号详细拆解：

- p：概率函数 (probability)。
- $|$：条件符号，读作“给定...的情况下”。
- s, a：**当前**的状态和动作（因）。
- s', r：**下一时刻**的状态和获得的奖励（果）。
- $\doteq$：定义为。
- **含义**：如果我现在处于状态 s，并且做了动作 a，那么下一刻变成状态 s' 且拿到奖励 r 的概率是多少？

关键性质：

因为它是概率分布，所以对于给定的 s 和 a，所有可能的结果概率之和必须为 1：

$$\sum_{s' \in S} \sum_{r \in \mathcal{R}} p(s', r|s, a) = 1$$

▼ 为什么做了动作a‘变成下一个状态S′且拿到奖励这一行为是概率分布？同时你怎么理解做出行为+拿到奖励这两个？是一回事嘛？还是做出行为不一定拿到相应的奖励？

这是一个非常敏锐的问题！你正在触及强化学习中**“环境（Environment）”**最底层的运作逻辑。

你提到的公式 $p(s', r | s, a)$ 描述的是**环境动力学 (Dynamics)**，也就是“这个世界运行的物理法则”。

我们把你的疑问拆解成三个核心层面来回答：

1. 为什么是“概率分布”？而不是确定的结果？

直觉：这个世界充满了不确定性。

在简单的游戏里（比如象棋），如果你把“卒”向前推一步，结果是确定的：卒一定会在前一格，盘面一定变成那样。这就是**确定性环境 (Deterministic Environment)**。

但在现实世界或复杂的强化学习问题中，事情往往不是确定的。这叫**随机性环境 (Stochastic Environment)**。

举个“投篮”的例子：

- **状态 (s)**：你站在三分线外，手里拿着球。
- **动作 (a)**：你以完美的姿势投出篮球。
- **结果**：
  - **情况 A (90%概率)**：球进了空心网。$\rightarrow$ **新状态 s'**: 得分牌变了，球在地上。**奖励 r**: +3分。
  - **情况 B (10%概率)**：一阵风吹过，球砸框弹出了。$\rightarrow$ **新状态 s''**: 得分牌没变，球飞远了。**奖励 r**: 0分。

结论：

即使你的动作 a 完全一样，环境反馈给你的结果（s', r）也可能有多种情况。

公式 $p(s', r | s, a)$ 就是在描述这张**“可能性的清单”**：

> “在状态 s 做动作 a，有 90% 的概率变成 s' 并得 3 分；有 10% 的概率变成 s'' 并得 0 分。”

---

2. 怎么理解“状态改变 + 拿到奖励”这两个？是一回事吗？

这两个是**绑定在一起的“套餐” (Joint Outcome)**，它们共同构成了“你做完动作后发生的事”。

- **$S_{t+1}=s'$ (下一刻的状态)**：告诉你**“你现在在哪？”**（环境变成了什么样）。
- **$R_{t+1}=r$ (奖励)**：告诉你**“刚才做得好不好？”**（刚才那个变动值多少分）。

为什么要把它们写在一个公式里 $p(s', r | \dots)$？

因为**奖励通常取决于你落到了哪个状态**。

例子：扫地机器人

- **动作**：向前冲。
- **可能结果 1**：成功冲进卧室（状态 $s_{bedroom}$），没撞墙。$\rightarrow$ 奖励 +1（因为打扫了新区域）。

- **可能结果 2**：轮子打滑，一头撞在墙上（状态 s_{crash}）。$\rightarrow$ 奖励 -10（因为撞坏了）。

你看，奖励（+1 还是 -10）与结果状态（进卧室还是撞墙）是**紧密耦合**的。你不能把它们完全割裂开看。公式里的逗号 s', r 表示这是一个**联合概率分布 (Joint Probability Distribution)**。

---

3. 是“做出行为不一定拿到相应的奖励”吗？

是的，准确地说是：**“做出行为，不一定拿到你‘预期’的那个奖励”。**

这包含了两层含义：

1. **随机性 (Stochasticity)**：

   就像上面的投篮例子，你觉得你投得很好（动作 a 对了），理应拿到 +3 分（预期奖励）。但因为环境有随机性（风吹、手滑），你实际拿到的可能是 0 分。

   - **公式表达**：$P(\text{reward}=0 \mid s, a) > 0$。

2. **稀疏性 (Sparsity)**：

   有时候环境设定就是这样：你做动作 a，环境确实发生了变化（$s \to s'$），但它觉得这事儿平平无奇，于是给你发了一个 **0** 分的奖励。

   - *注意*：在数学上，**0 也是一个奖励**。系统**总是**会返回一个数值 r，只不过有时候 $r=0$，有时候 r 是随机波动的数值。

总结图解

为了更直观，我们可以把这个过程画成一个分支图：

代码段

```
graph TD
    Start((当前状态 S)) -->|动作 A| Outcomes{环境判定}
    Outcomes -->|概率 0.8| Result1[结果 1]
    Outcomes -->|概率 0.2| Result2[结果 2]

    subgraph Result1 [结果套餐 1]
    S1(新状态 S')
    R1(奖励 r = +10)
    end

    subgraph Result2 [结果套餐 2]
```

```
    S2(新状态 S'')
    R2(奖励 r = -5)
    end
```

你的理解修正：

- 不是"不一定拿到奖励"，而是**"一定会拿到一个奖励信号（哪怕是0），但这个奖励数值的大小以及你会去到的新状态，是服从一个概率分布的"**。
- s' 和 r 是环境对你动作的**双重反馈**——一个告诉你处境变了（物理变化），一个告诉你得分变了（价值评价）。

3. 衍生概念推导

有了核心动力学 $p(s', r|s, a)$，我们可以推导出其他有用的概率：

1. **状态转移概率 (State-transition probabilities)**：

   我不关心拿多少分，只关心做完动作 a 后跳到状态 s' 的概率。

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

   推导思路：这是概率论中的**边缘化 (Marginalization)**，把所有可能的奖励 r 加起来消除掉。

2. **期望奖励 (Expected rewards)**：

   在状态 s 做动作 a，平均能拿多少分？

   $r(s, a) = \mathbb{E}[R_{t+1}|S_t = s, A_t = a] = \sum_{r \in \mathcal{R}} \sum_{s' \in S} r \cdot p(s', r|s, a)$

   推导思路：这是**期望 (Expectation)** 的定义，即 $\sum(\text{值} \times \text{概率})$。

第二部分：目标与回报 (Returns)

1. 直觉：短视 vs. 远视

Agent 的目标是最大化它在整个过程中获得的总奖励。但是，今天的 100 块钱通常比明年的 100 块钱更值钱。在 RL 中，我们也引入这种机制。

2. 任务分类

- **情节性任务 (Episodic Tasks)**：
  - 有明确的终点（例如：一局围棋、一次迷宫跑酷）。

- 序列会终止于时间 T。
- 之后系统重置，开始独立的新一局。

- **持续性任务 (Continuing Tasks)**：
  - 永不停止（例如：恒温器调节温度、机器人持续行走）。
  - $T = \infty$。

3. 回报 (Return, G_t) 的定义

我们要最大化的不是某一步的奖励 R，而是未来的**累积回报** G_t。

- **无折扣回报 (简单求和)**：

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

  这适用于情节性任务。

- **折扣回报 (Discounted Return)**：

  引入折扣率 γ (gamma)，$0 \le \gamma \le 1$。

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

  - 如果 $\gamma = 0$：**短视 (Myopic)**，只关心眼前的奖励 R_{t+1}。
  - 如果 $\gamma \to 1$：**远视 (Farsighted)**，未来的奖励和现在的奖励看得一样重。

4. 关键数学推导：为什么用折扣？

1. **数学收敛性**：对于持续性任务（无限步），如果奖励不衰减，总和可能是无穷大，无法比较。如果 $\gamma < 1$ 且奖励是有上限的（设最大奖励为 u），那么回报也是有限的：

   $|G_t| \le \sum_{k=0}^{\infty} \gamma^k u = u \sum_{k=0}^{\infty} \gamma^k = \frac{u}{1-\gamma}$

   (这是等比数列求和公式)

2. **递归性质 (Recursive Property)**：

   这是强化学习中最著名的公式推导之一，它将现在的回报与未来的回报联系起来：

   $$\begin{aligned} G_t &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \dots \\ &= R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \dots) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned}$$

这意味着：**今天的总身价 = 今天赚的钱 + 0.9 × 明天的总身价。**

策略（Policy）与价值函数（Value Functions），以及连接现在与未来的**贝尔曼方程（Bellman Equation）**。

第一部分：核心概念的直觉理解

1. 策略 (Policy, π) —— 你的“游戏攻略”

- **直觉**：你在玩游戏时，脑子里有一套规则：“看到敌人（状态 s），我就开枪（动作 a）”；或者“看到红灯（状态 s），我就停下（动作 a）”。这就叫策略。
- **数学**：它是从**状态**到**动作概率**的映射。

2. 价值函数 (Value Function) —— 这一步“值”多少钱？

- **直觉**：
  - 在下围棋时，虽然还没赢，但你看一眼棋盘说：“这局势我稳了（胜率90%）”。这个“90%”就是当前状态的**价值 (V)**。
  - 如果你在这一步决定“下天元”，你心里会评估：“这步棋下在天元虽然现在没吃子，但长远看能占优”。这就是**动作价值 (Q)**。
- **核心区别**：
  - $V(s)$：评估这个**位置**好不好（比如站在悬崖边，不管下一步咋样，这位置就很危险，分低）。
  - $Q(s, a)$：评估在这个位置**做某件事**好不好（站在悬崖边是 s，但如果你动作 a 是“往回退”，那这个动作的价值就很高）。

3. 贝尔曼方程 (Bellman Equation) —— 价值的递归

- **直觉**：这一步的价值，等于“**眼前的甜头**”加上“**未来的潜力**”。
- **通俗公式**：

  $$\text{今天的身价} = \text{今天的收入} + (\text{折扣率} \times \text{明天的身价})$$

  这种“把大问题拆成当前一步 + 剩余问题”的思路，就是贝尔曼方程的核心。

第二部分：数学符号与公式深度拆解

1. 策略 (Policy)

符号：$\pi(a|s)$

- **含义**：在状态 s 下，选择动作 a 的**概率**。
- **例子**：$s =$ 十字路口。
  - $\pi(\text{向左}|s) = 0.2$
  - $\pi(\text{向右}|s) = 0.8$
  - (加起来必须等于 1)。

2. 状态价值函数 (State-Value Function, V^π)

公式：

$V^\pi(s) = \mathbb{E}_\pi[G_t|S_t = s]$

- **拆解**：
  - $V^\pi(s)$：当我也按照策略 π 玩的时候，状态 s 的价值。
  - $\mathbb{E}_\pi$：**期望值 (Expectation)**。为什么要算期望？因为未来是不确定的（你可能按错键，环境可能有风，下一关的奖励不一样），所以我们要算所有可能情况的平均值。
  - G_t：**回报 (Return)**。就是我们上一节学的“未来所有奖励的折扣总和”。
- **人话翻译**：如果我从状态 s 开始，一直按攻略 π 玩下去，平均能拿多少分？

3. 动作价值函数 (Action-Value Function, Q^π)

公式：

$Q^\pi(s, a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$

- **拆解**：注意这里条件变成了 $S_t = s, A_t = a$。
- **人话翻译**：如果我从状态 s 开始，**强制先做动作** a（哪怕攻略不推荐），然后**之后**都按攻略 π 玩，平均能拿多少分？

- **重要性**：这就是 Q-Learning 的核心，它是用来选动作的。比较 $Q(s, \text{左})$ 和 $Q(s, \text{右})$，哪个大选哪个。

4. 贝尔曼方程 (Bellman Equation) —— 最难但也最重要的部分

公式 (3.10)：

$V^{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r + \gamma V^{\pi}(s') \right]$

让我们像剥洋葱一样拆解这个"恐怖"的公式：

1. **最里面** $[r + \gamma V^{\pi}(s')]$：

   这是**"单次交互的价值"**。

   - r：刚才那一步拿到的**即时奖励**。
   - $\gamma V^{\pi}(s')$：打折后的**下一站身价**。
   - 合起来就是：走到下一步的总价值。

2. **中间层** $\sum_{s',r} p(s', r|s, a)(\dots)$：

   这是**"环境的不确定性"**。

   - 既然做动作 a 之后，可能会变成 s'，也可能变成别的，环境通过概率 p 来控制。
   - 这里把所有可能的下一站结果加权平均。

3. **最外层** $\sum_{a \in \mathcal{A}} \pi(a|s)(\dots)$：

   这是"**你的不确定性**"。

   - 在状态 s，你可能选动作 A，也可能选动作 B。
   - 这里把每种动作带来的后果，按照你选择该动作的概率 $\pi(a|s)$ 进行加权平均。

- **三个组成部分**:
  1. **即时奖励**: r
  2. **未来价值**: $\gamma V^{\pi}(s')$ (打折后的下一状态价值)
  3. **加权平均**: 这里的两个求和号 ($\sum$) 分别对**策略的选择**和**环境的跳转**求期望。
  4. 为何是加权平均？：因为pi 是0-1的数，属于选择动作的策略概率

总结：当前身价 = (我可能做的动作 $\times$ (环境可能给的反馈 + 下一站的身价)) 的总平均。

---

4. 关键联系

- **V 和 Q 的关系**:

  状态价值 V 其实就是动作价值 Q 的平均值（基于策略概率加权）：

  $V^{\pi}(s) = \sum_{a} \pi(a|s) Q^{\pi}(s,a)$

  (这句话的意思是：这一关好不好打，取决于我在这一关能采取的所有动作的平均效果)。

这部分内容是强化学习的**终极目标**——寻找**最优策略 (Optimal Policy)** 和 **最优价值 (Optimal Value)**。

如果说上一节是“给你一本攻略，算算你能得多少分”，这一节就是“**别给我攻略，告诉我怎么玩才能得最高分！**”。

这是强化学习从“评估”走向“控制”的关键一步。

第一部分：核心概念的直觉理解

1. 什么是“最优”？

- **直觉**：
  - 你和你朋友都在玩《超级马里奥》。
  - 你的策略 (π) 是“见到怪就躲”，平均能玩到第 3 关。
  - 你朋友的策略 (π') 是“见到怪就踩”，平均能玩到第 8 关。
  - 显然，你朋友的策略比你好 ($\pi' > \pi$)。
  - **最优策略 (π_*)** 就是：这世界上存在一种完美的玩法，无论你在哪一关，按它的玩，得分都是理论上最高的，不可能有人比它更高。

2. 最优价值函数 (v_* 和 q_*) —— 上帝视角的评分表

- $v_*(s)$：在状态 s，如果你用**最完美**的技术玩，以后一共能拿多少分？（这是这个状态的**理论上限**）。

- $q_*(s,a)$：在状态 s，如果你先强制做动作 a，然后后面全部用**最完美**的技术玩，能拿多少分？

3. 这里的魔法 —— 把“长期”变成“眼前”

- 如果你拥有了这张“上帝评分表” (v_*)，你就不需要像预言家一样预测未来 100 步了。
- 你只需要看眼前的一步：哪个动作能让你跳到一个“分最高”的格子，你就选哪个。
- **结论**：只要算出了最优价值，**贪婪策略（Greedy Policy）就是最优策略**。

第二部分：数学符号与公式深度拆解

1. 策略的好坏比较 (Partial Ordering)

怎么定义策略 π 比 π' 好？

$$\pi \geq \pi' \iff v^{\pi}(s) \geq v^{\pi'}(s), \quad \forall s \in S$$

- **含义**：必须在**每一个**状态下，π 的得分都不低于 π'，才能说 π 更好。

2. 最优价值函数 (Optimal Value Functions)

我们定义 $v_*(s)$ 为所有可能策略中能达到的最大值：

$$v_*(s) = \max_{\pi} v^{\pi}(s)$$

同理，$q_*(s,a)$ 也是取最大值：

$$q_*(s,a) = \max_{\pi} q^{\pi}(s,a)$$

3. 贝尔曼最优方程 (Bellman Optimality Equation) —— 关键变化

这是本节最重要的公式。请注意它和上一节的**区别**：上一节是求平均 ($\sum \pi$)，这一节是**挑最好的 ($\max$)**。

状态价值的最优方程：

$$v_*(s) = \max_{a \in \mathcal{A}} q_*(s,a) = \max_{a} \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')]$$

- **拆解**：
  - $v_*(s) = \max_a \dots$：这意味着状态 s 的价值，等于它所有能做的动作里，**那个结果最好的动作**带来的价值。

- **不需要策略概率 $\pi(a|s)$ 了**：因为我们要找最优解，所以我们只关心那个能带来最大收益的动作，其他的“烂操作”直接无视。

动作价值的最优方程：

$q_*(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma \max_{a'} q_*(s',a')]$

- **拆解**：
  - 我现在做了动作 a（哪怕它不是最好的）。
  - 环境把我送到了 s'。
  - $\max_{a'} q_*(s',a')$：到了 s' 之后，因为我是高手，我一定会选在那边能拿最高分的动作 a'。
  - 这就是递归的精髓：**现在的最优 = 即时奖励 + 未来的最优**。

---

4. 为什么 v_* 如此重要？

一旦我们解出了 v_*，寻找最优策略就变得非常简单（甚至不需要“寻找”）：

- **贪婪即最优**: 对于每个状态 s，你只需要看哪个动作 a 能让 $r + \gamma v_*(s')$ 最大，直接选它就行。
- **短视变远视**: $v_*(s')$ 已经包含了未来所有步骤的最优回报。所以，通过只看“下一步”+“下一站的最优价值”，你就相当于看透了全局。

5. 现实的困难

虽然公式很完美，但直接解这个方程（例如穷举搜索）在现实中往往**太慢了**或者**不可行**。

- **计算昂贵**: 状态太多。
- **模型未知**: 我们往往不知道环境动力学 $p(s',r|s,a)$ 到底是什么。
- **解决方法**: 这就是为什么我们需要 **Q-Learning**, **DQN** 等算法——它们是用来**近似**求解这个方程的工具。