绪论

在正式介绍具体的强化学习（reinforcement learning，RL）算法之前，本章将从宏观角度讨论一下强化学习的相关概念以及应用等，帮助读者更好地“观其大略”。尤其是对于想用利强化学习做一些交叉研究的读者来说，更应该先通过本章了解强化学习是什么、大概能做什么、能够实现什么样的效果等，而不是直接从一个个具体的算法开始学习。

强化学习发展至今，虽然算法已经有成百上千种样式，但实际上从大类来看要掌握的核心算法并不多，大多数算法都只是在核心算法的基础上做了一些较小的改进。举个例子，如图 1 所示，我们知道水加上咖啡豆通过一定的方法就能调制成咖啡，水加上糖块就能变成糖水，它们虽然看起来形式不同，但本质上都是水，只是有不同的口味而已。

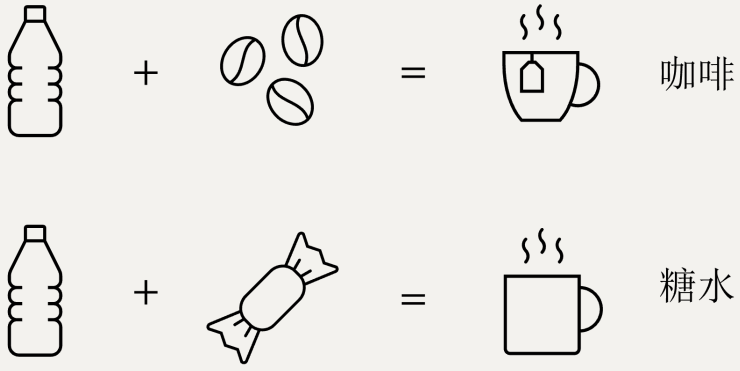

图 1: 咖啡与糖水的示例

为什么要学习强化学习？

我们先讨论一下为什么要学习强化学习，以及强化学习对于我们的意义。可能大部分读者都是通过了解人工智能才了解到强化学习的，但实际上早在我们认识人工智能之前可能就已经不知不觉地接触到了强化学习。

笔者想起了初中生物课本中关于蚯蚓的一个实验，其内容大致是这样的：如图 2 所示，将蚯蚓放在一个盒子中，盒子中间有一个分叉路口，路的尽头分别放有食物和电极，让蚯蚓自己爬行到其中一条路的尽头，在放有食物的路的尽头蚯蚓会品尝到食物的美味，而在放有电极的路的尽头则会受到轻微的电击。

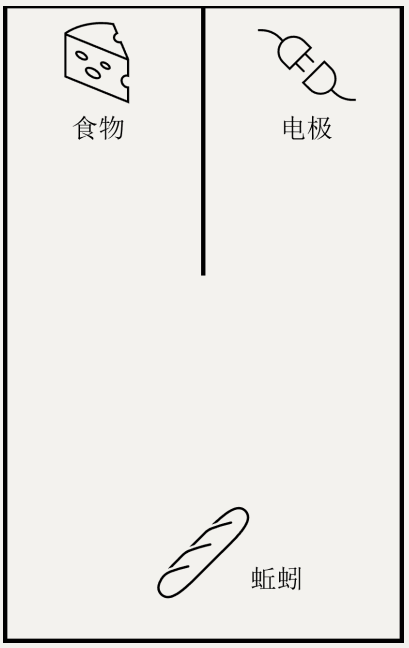

图 2: 蚯蚓实验

该实验的目的是让蚯蚓能一直朝着有食物的路爬行，但由于蚯蚓没有真正的眼睛，因此一开始蚯蚓可能会一直朝着有电极的路爬行并且遭到电击。每次蚯蚓遭到电击或者吃到食物之后实验人员会将其放回原处，经过多次实验，蚯蚓会逐渐学会朝着有食物的路爬行，而不是朝着有电极的路爬行。

在这个过程中，蚯蚓在不断地尝试和试错中学习到了正确的策略，虽然初中生物课本中这个实验的目的是为了说明蚯蚓的运动是由外界刺激所驱动的，而不是蚯蚓自身的意志所驱动的，但在今天，从人工智能的角度来看，这其实带有较为鲜明的强化学习的“味道”，即试错学习（ trial and error learning ）。

试错学习一开始是和行为心理学等工作联系在一起的，主要包括以下几个关键部分：

- 尝试：采取一系列动作或行为来尝试解决问题或实现目标。
- 错误：在尝试的过程中可能会出现错误，这些错误可能是环境的不确定性导致的，也可能是自身的不当行为导致的。
- 结果：每次尝试的后果，无论是积极的还是消极的，都会对下一次尝试产生影响。
- 学习：通过不断地尝试并出现错误，自身会逐渐积累经验，了解哪些动作或行为会产生有利的结果，从而在下一次尝试中做出更加明智的选择。

试错学习在我们的日常生活中屡见不鲜，并且通常与其他形式的学习形成对比，例如经典条件反射（巴甫洛夫条件反射）和观察学习（通过观察他人来学习）。注意，试错学习虽然是强化学习中最鲜明的要素之一，但并不是强化学习的全部，强化学习还包含其它的学习形式例如观察学习（对应模仿学习、离线强化学习等技术）。

此外，在学习过程中个人做出的每一次尝试都是是一次决策（decision），每一次决策都会带来相应的后果，这个后果可能是好的，也可能是坏的，也可能是即时的，比如我们吃到棉花糖就能立刻感受到它的甜度，也可能是延时的，比如寒窗苦读十年之后，方得一日踏阅长安花。

我们把好的结果称为奖励（reward），坏的结果称为惩罚（punishment）或者负的奖励。最终通过一次次的决策来实现目标，这个目标通常是以最大化累积的奖励来呈现的，这个过程就是序列决策（sequential decision making）过程，而强化学习就是解决序列决策问题的有效方法之一，即本书的主题。换句话说，对于任意问题，只要能够建模成序列决策问题或者带有鲜明的试错学习特征，就可以使用强化学习来解决，并且这是截至目前最为高效的方法之一，这就是要学习强化学习的原因。

强化学习的应用

从 1.1 中我们了解了强化学习大概是用来做什么的，那么它能实现什么样的效果呢？本节我们就来看看强化学习的一些例子和实际应用。强化学习的应用场景非常广泛，其中最为典型的场景之一就是游戏，以 AlphaGo 为代表的围棋 AI 就是强化学习的代表作之一，也是其为人们广泛熟知的得意之作。除了部分棋类游戏，以 AlphaStar 为代表的《星际争霸》AI、以 AlphaZero 为代表的通用游戏 AI，以及近年以 OpenAI Five 为代表的 Dota 2 AI，这些都是强化学习在游戏领域的典型应用。

除了游戏领域之外，强化学习在机器人领域中也有所应用。举个例子，图 3 演示了 NICO 机器人学习抓取任务的过程。该任务的目标是将桌面上的物体抓取到指定的位置，机器人通过每次输出相应关节的参数来活动手臂，然后通过摄像头观测当前的状态，最后通过人为设置的奖励（例如接近目标就给一个奖励）来学习到正确的抓取策略。

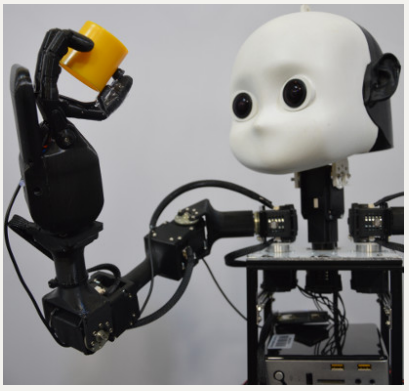

图 3: NICO 机器人学习抓取任务

不同于游戏领域，在机器人中实现强化学习的成本往往较高，一方面观测环境的状态需要大量的传感器，另一方面则是试错学习带来的实验成本，在训练过程中如果机器人决策稍有失误就有可能导致设备损坏，因此在实际应用中往往需要结合其他的方法来辅助强化学习进行决策。其中最典型的方法之一就是建立一个仿真环境，通过仿真环境来模拟真实环境，这样就可以大大降低实验成本。

如图 4 所示，该仿真环境模拟了机器人抓取任务的真实环境，通过仿真环境免去大量视觉传感器的搭建过程从而可以大大降低实验成本，同时由于仿真环境中机器人关节响应速度更快，进而算法的迭代速度更快，可以更快地得到较好的策略。

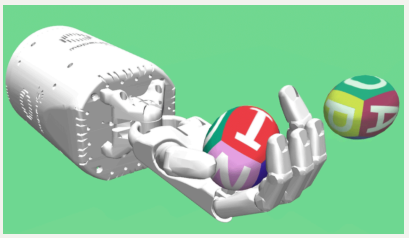

图 4: 机器人抓取任务的仿真环境

当然，仿真环境也并不是万能的，因为仿真环境和真实环境之间往往存在一定的差异，这就需要我们在设计仿真环境的时候尽可能全面地考虑到真实环境的各种因素，这也是一个非常重要的研究方向。除了简单的抓取任务之外，研究者们还在探索将强化学习应用于更加复杂的机器人任务，例如仓储搬运、机器人足球以及自动驾驶等等。

除了游戏和机器人领域之外，强化学习在金融领域也有所应用，例如股票交易、期货交易、外汇交易等。在股票交易中，我们的目标是通过买卖股票来最大化我们的资产。在这个过程中，我们需要不断地观测当前的股票价格，然后根据当前的价格来决定买入或卖出股票的数量，最后通过股票价格的变化来更新我们的资产。在这个过程中，我们的资产会随着股票价格的变化而变化，这就是奖励或惩罚，每次的买卖就是决策。当然，强化学习的应用还远远不止如此，例如自动驾驶、推荐系统、交通派单、广告投放以及近来大火的 ChatGPT 等，这些都是。

强化学习方向概述

强化学习不仅应用十分广泛，而且从技术角度来讲其方向也非常多。在学习基础的强化学习知识之后，读者可根据自身的兴趣选择相应的方向进行深入学习。本节将对强化学习的一些典型方向进行简要介绍，以便读者能够对强化学习有更加全面的认识，同时也为后续的学习做好铺垫，强化学习的典型应用主要如下。

多智能体强化学习

顾名思义，多智能体强化学习（multi-agent reinforcement learning，MARL）就是在多个智能体的环境下进行强化学习。与单智能体环境不同，在多智能体环境中通常存在非静态问题，即环境的状态不仅由智能体的动作决定，还受到其他智能体的动作的影响。例如在 AlphaStar 中，每个智能体都是一个《星际争霸》中的玩家，每个玩家都有自己的目标，例如攻击对方的基地或者防守自己的基地，这就导致环境的状态不仅由自己的动作决定，还受到其他玩家的动作的影响。

其次存在信号问题，即智能体之间可能需要进行通信以合作或竞争，如何高效地通信并从信号中学习是一个难题。还存在信誉分配问题，在多智能体的合作任务中，确定每个智能体对于整体目标的贡献（或责任）是一个挑战。多智能体环境通常也存在复杂的博弈场景，对于此类研究往往引入博弈论来找到环境中的纳什均衡或其他均衡策略，但这是一个复杂的挑战。

从数据中学习

从数据中学习，或者从演示中学习（learn from demostration）包含丰富的门类，例如以模仿学习为代表的从专家数据中学习策略、以逆强化学习（inverse reinforcement learning，IRL 为代表的从人类数据中学习奖励函数和以及从人类反馈中学习（reinforcement learning from human feedback，RLHF）为代表的从人类标注的数据中学习奖励模型来进行微调（fine-tune）。此外，还包括离线强化学习（offline reinforce learning）、世界模型（world model）等等，这些方法都利用了数据来辅助强化学习，因此本书将它们同归为一类从数据中学习的方法。注意，这些方法的思路实际上是大相径庭的，完全可以作为一个单独的子方向来研究。

模仿学习是指在奖励函数难以明确定义或者策略本身就很难学出来的情况下，我们可以通过模仿人类的行为来学习到一个较好的策略。最典型的模仿策略之一就是行为克隆（behavioral cloning，BC），即将每一个状态-动作对视为一个训练样本，并使用监督学习的方法（如神经网络）来学习一个策略。但这种方法容易受到分布漂移（distribution shift）的影响。智能体可能会遇到从未见过的状态，导致策略出错。

逆强化学习是指通过观察人类的行为来学习到一个奖励函数，然后通过强化学习来学习一个策略。由于需要专家数据，逆强化学习会受到噪声的影响，因此如何从噪声数据中学习到一个较好的奖励函数也是一个难题。

探索策略

探索策略（exploration strategy）。在强化学习中，探索策略是一个非常重要的问题，即如何在探索和利用之间做出权衡。在探索的过程中，智能体会尝试一些未知的动作，从而可能会获得更多的奖励，但同时也可能会遭受到惩罚。而在利用的过程中，智能体会选择已知的动作，从而可能会获得较少的奖励，但同时也可能会遭受较少的惩罚。因此，如何在探索和利用之间做出权衡是一个非常重要的问题。目前比较常用的方法有$\epsilon-$greedy和置信上界（upper confidence bound，UCB）等等。此外，提高探索的本质也是为了避免局部最优问题，从而提高智能体的鲁棒性，近年来也有研究结合进化算法来提高探索的效率，例如 NEAT（neuro evolution of augmenting topologies）和 PBT（population based training）等算法，当然这些算法在提高探索的同时也会带来一定的计算成本。

实时环境

实时环境（ real-time environment ）。在实际应用中，智能体往往需要在实时或者在线环境中进行决策，例如自动驾驶、机器人等等。在这种情况下训练不仅会降低效率（实时环境响应动作更慢），而且还会带来安全隐患（训练过程中可能会出现意外）。解决这一问题的思路之一就是离线强化学习（ offline reinforcement learning ），即在离线环境中进行训练，然后将训练好的模型部署到在线环境中进行决策。但这种方法也存在着一定的问题，例如离线环境和在线环境之间可能存在着分布漂移，即两个环境的状态分布不同，这就导致了训练好的模型在在线环境中可能会出现意外。另外有一种是近两年比较流行的思路，即世界模型（ world model ），即在离线环境中训练一个世界模型，然后将世界模型部署到在线环境中进行决策。世界模型的思路是将环境分为两个部分，一个是世界模型，另一个是控制器。世界模型的作用是预测下一个状态，而控制器的作用是根据当前的状态来决策动作。这样就可以在离线环境中训练世界模型，然后将世界模型部署到在线环境中进行决策，从而避免了在线环境中的训练过程，提高了效率，同时也避免了在线环境中的安全隐患。但世界模型也存在着一定的问题，例如世界模型的预测误差会导致控制器的决策出错，因此如何提高世界模型的预测精度也是一个难题。

多任务强化学习

多任务强化学习（ multi-task reinforcement learning ）。这个问题在深度学习中也较为常见，在实际应用中，智能体往往需要同时解决多个任务，例如机器人需要同时完成抓取、搬运、放置等任务，而不是单一的抓取任务。在这种情况下，如何在多个任务之间做出权衡是一个难题。目前比较常用的方法有联合训练（ joint training ）和分层强化学习（ hierarchical reinforcement learning ）等等。联合训练的思路是将多个任务的奖励进行加权求和，然后通过强化学习来学习一个策略。分层强化学习的思路是将多个任务分为两个层次，一个是高层策略，另一个是低层策略。高层策略的作用是决策当前的任务，而低层策略的作用是决策当前任务的动作。这样就可以通过强化学习来学习高层策略和低层策略，从而解决多任务强化学习的问题。但分层强化学习也存在着一定的问题，例如高层策略的决策可能会导致低层策略的决策出错，因此如何提高高层策略的决策精度也是一个难题。

学习本书之前的一些准备

我们先介绍一下关于本书的初衷。其实目前强化学习相关的书籍在市面上已经琳琅满目了，但是这些普遍偏向理论，缺少一些实际的经验性总结，比如大佬们可能会通过数学推导来告诉你某某算法是可行的，可是一些实验细节和不同算法的对比很难在这些书籍中体现出来，理论与实践之间、公式与代码之间其实存在着一定的鸿沟。另一方面，由于信息时代知识的高速迭代，面对如海洋一般的信息，我们需要从中梳理出重点并快速学习，以便于尽快看到实际应用的效果，而这中间就不得不需要一个经验丰富的老师傅来带路了，这也是本书的初衷之一。笔者会基于大量的强化学习实践经验，对于理论部分删繁就简，并与实践紧密结合，以更通俗易懂的方式帮助读者们快速实践。

其次，在具体的学习之前，先给读者做一些基础的知识铺垫。第一，强化学习是机器学习的一个分支，因此读者需要具备一定的机器学习基础，例如基本的线性代数、概率论、数理统计等等。当然只需要读者们修过相关的大学课程即可，不必先去刻意回顾一些知识，原理部分跟随本书的推导即可。第二，在学习强化学习初期是不涉及深度神经网络相关的东西的，这一部分通常称为传统强化学习部分。尽管这部分的算法在今天已经不常用，但是其中蕴含的一些思想和技巧是非常重要的，因此读者们需要对这部分内容有所了解。在过渡到结合神经网络的深度强化学习部分之前，本书会花一章的时间帮助读者整理需要的深度学习知识。

深度学习在强化学习中扮演的角色主要是提供了一个强大的函数拟合能力，使得智能体能够处理复杂、高维度和非线性的环境。深度学习与强化学习之间的关系相当于眼睛和大脑的关系，眼睛是帮助大脑决策更好地观测世界的工具，对于一些没有眼睛的动物例如蚯蚓也可以通过其他的感官来观测并解析状态。再比如，同样大脑水平下，即相同的强化学习算法条件下，正常人要比双目失明的人日常的决策要更方便。但是，即使深度学习部分是相同的，例如正常大人和小孩都能通过眼睛观测世界，然由于大脑决策水平的差异也会让两者表现有所差异。总而言之，深度与强化在更复杂的环境下缺一不可。最后，尽管强化学习算法很多，但基本上就分为两类，即基于价值的和基于策略梯度的算法，这两种算法各有优势，读者们在学习之后根据实际需要选择即可。

马尔可夫决策过程

本章开始介绍马尔可夫决策过程的基本概念，包括马尔可夫性质、回报、状态转移矩阵等内容。马尔可夫决策过程是强化学习的核心问题模型，即想用强化学习来解决问题，首先需要将问题建模为马尔可夫决策过程，并明确状态空间、动作空间、状态转移概率和奖励函数等要素。此外，还介绍了策略、状态价值和动作价值等重要概念，这些概念在后续的强化学习算法中会频繁用到，务必牢记。

马尔可夫决策过程

在强化学习中，马尔可夫决策过程（Markov Decision Process, MDP）是用来描述智能体与环境交互的数学模型。如图 1 所示，智能体（Agent）与环境（Environment）在一系列离散的时步（ time step ）中交互，在每个时步 t，智能体接收环境的状态 s_t，并根据该状态选择一个动作 a_t。执行该动作后，智能体会收到一个奖励 r_t，同时环境会转移到下一个状态 s_{t+1}。

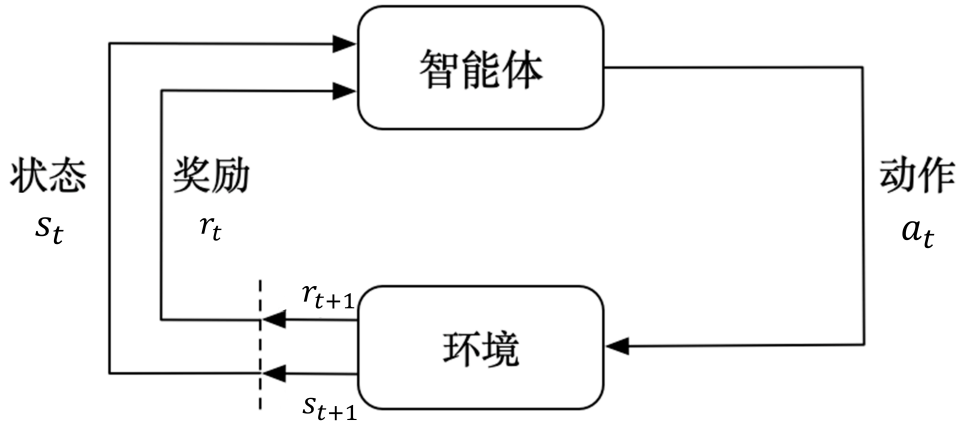

图 1: 智能体与环境的交互过程

这个过程不断重复，形成一条**轨迹**，如式 (1) 所示。

$$s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_t, a_t, r_t, \cdots \tag{1}$$

完成一条完整的轨迹，即从初始状态到终止状态（例如玩游戏玩到最终的胜负结算阶段），也称为一个**回合**（ episode ），通常在有限的时步 T 后结束，即 $t = 0, 1, 2, \cdots, T$，T 是回合的最大步数。

如果要用强化学习来解决问题，首先需要将问题建模为马尔可夫决策过程，即明确状态空间、动作空间、状态转移概率和奖励函数等要素。通常我们用一个五元组来定义马尔可夫决策过程，如式 (2) 所示。

$$MDP = (S, A, P, R, \gamma) \tag{2}$$

其中 S 是状态空间，表示所有可能的环境状态的集合，A 是动作空间，表示智能体可以选择的所有可能动作的集合，P 是状态转移概率矩阵，描述了在给定当前状态和动作的情况下，环境转移到下一个状态的概率分布，R 是奖励函数，定义了在特定状态下执行某个动作所获得的即时奖励，γ 是折扣因子，用于权衡当前奖励和未来奖励的重要性，其取值范围在 0 到 1 之间。其中状态转移矩阵和折扣因子将在下文详细展开说明。

马尔可夫性质

马尔可夫决策过程的核心假设是**马尔可夫性质（Markov Property）**，即系统未来状态的概率分布只依赖于当前的状态和动作，而与过去的状态和动作无关，如式 (3) 所示。

$$P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \cdots, s_0, a_0) = P(s_{t+1}|s_t, a_t) \tag{3}$$

然而，在真实世界中严格满足马尔可夫性质的情况并不多见，但大多情况下，我们依然可以通过适当的状态表示来近似满足马尔可夫性质。

例如在自动驾驶中，将当前车辆的位置、速度和周围环境信息作为状态表示，这些信息足以预测下一时刻的状态，而不需要考虑更早之前的历史数据，从而近似满足马尔可夫性质，这样的过程也叫做**部分可观测马尔可夫决策过程（Partially Observable Markov Decision Process, POMDP）**。

状态转移矩阵

通常，马尔可夫决策过程通常指有限马尔可夫决策过程（finite MDP），即状态空间和动作空间都是有限的。如果状态空间或动作空间是无限的，通常需要采用其他方法进行建模，例如连续时间马尔可夫过程等。

既然状态数有限，就可以用一种状态流向图的形式表示智能体与环境交互过程中的走向。如图 2 所示，图中每个曲线箭头表示指向自己，对于状态 s_1 来说，有 0.2 的概率继续保持在 s_1 状态，同时也有 0.4 和 0.4 的概率转移到状态 s_2 和 s_3。同理，其他状态之间也有类似的转移概率。

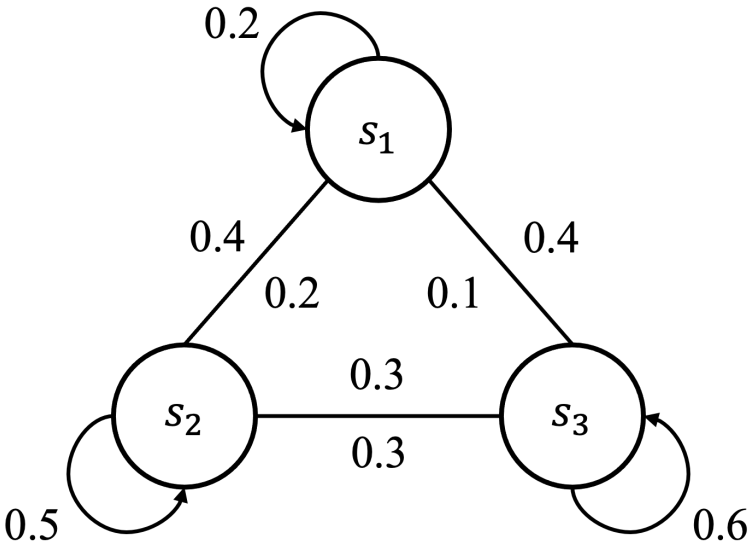

图 2: 马尔可夫链

注意，图 2 中并没有包含动作和奖励等元素，因此严格来说它表示的是**马尔可夫链（Markov Chain）**，又叫做离散时间的马尔可夫过程（Markov Process），但它与马尔可夫决策过程有着密切的联系，都是基于马尔可夫性质构建的。

我们用一个概率来表示状态之间的切换，如式 (4) 所示。

$$P_{ss'} = P(S_{t+1} = s' | S_t = s) \tag{4}$$

即当前状态是 s 时，下一个状态是 s'的概率，其中大写的 S 表示所有状态的集合，即 $S = \{s_1, s_2, s_3\}$。例如，$P_{12} = P(S_{t+1} = s_2 | S_t = s_1) = 0.4$ 表示当前时步的状态是 s_1，下一个时步切换到 s_2 的概率为 0.4。

拓展到所有状态，可以把这些概率绘制成一个状态转移表，如表 1 所示。

表 1: 马尔可夫状态转移表

$S_t = s_1$	0.2	0.4	0.4
$S_t = s_2$	0.2	0.5	0.6

	$S_{t+1}=s_1$	$S_{t+1}=s_1$	$S_{t+1}=s_3$
$S_t=s_3$	0.1	0.3	0.6

在数学上也可以用矩阵来表示，如式 (5) 所示。

$$P_{ss'}=\begin{bmatrix}0.2 & 0.4 & 0.4\\ 0.2 & 0.5 & 0.3\\ 0.1 & 0.3 & 0.6\end{bmatrix} \tag{5}$$

这个矩阵就叫做**状态转移矩阵（State Transition Matrix）**，拓展到所有状态可表示为式 (6) 所示。

$$P_{ss'}=\begin{pmatrix}p_{11} & p_{12} & \cdots & p_{1n}\\ p_{21} & p_{22} & \cdots & p_{2n}\\ \vdots & \vdots & \ddots & \vdots\end{pmatrix} \tag{6}$$

其中 n 表示状态数，注意从同一个状态出发转移到其他状态的概率之和是等于 1 的，即 $\sum_{j=1}^{n}p_{ij}=1$，$i=1,2,\cdots,n$。**状态转移矩阵是环境的一部分**，描述了环境状态之间的转移关系。

目标与回报

在强化学习中，智能体的目标是**通过与环境的交互，学习一个最优策略，使得在每个状态下选择的动作能够最大化累积的奖励**。这个累积的奖励通常被称为**回报（Return）**，如式 (7) 所示。

$$G_t=R_{t+1}+\gamma R_{t+2}+\gamma^2R_{t+3}+\cdots=\sum_{k=0}^{\infty}\gamma^kR_{t+k+1} \tag{7}$$

表示从时间步 t 开始，未来所有奖励的加权和，其中 γ 是折扣因子，在 0 到 1 之间。折扣因子的作用是用来控制未来奖励在当前决策中的重要性。当 γ 接近 0 时，智能体更关注当前的奖励，而忽略未来的奖励；当 γ 接近 1 时，智能体会更加重视未来的奖励。

折扣因子一方面在数学上确保回报 G_t 的收敛性，另一方面也代表了时间价值，就像经济学中的货币折现，同样面值的货币现在拿到手中比未来拿到更有价值。此外，折扣因子还可以用来衡量智能体对长期回报的关注度，或者者说"能看到多远"，称之为有效视界（effective horizon），如式 (8) 所示。

$$H_{eff}=\frac{1}{1-\gamma} \tag{8}$$

当 $\gamma=0.9$ 时，$H_{eff}=10$，表示智能体主要关注未来 10 个时步内的奖励；当 $\gamma=0.99$ 时，$H_{eff}=100$，表示智能体关注未来 100 个时步内的奖励。

此外，当前时步的回报 G_t 跟下一个时步 G_{t+1} 的回报是有所关联的，即递归地定义，如式 (9) 所示。

$$\begin{aligned}G_t&\doteq r_{t+1}+\gamma r_{t+2}+\gamma^2r_{t+3}+\gamma^3r_{t+4}+\cdots\\&=r_{t+1}+\gamma\left(r_{t+2}+\gamma r_{t+3}+\gamma^2r_{t+4}+\cdots\right)\\&=r_{t+1}+\gamma G_{t+1}\end{aligned} \tag{9}$$

策略与价值

策略

策略 （Policy）表示智能体在每个状态下选择动作的规则或方法，用 π 表示。如式 (10) 所示，策略是一个从状态到动作的映射或函数。

$$\pi(a|s) = P(A_t = a|S_t = s) \tag{10}$$

表示在状态 s 下选择动作 a 的概率分布。**策略可以是确定性的（deterministic），即在每个状态下总是选择同一个动作，或者是随机性的（stochastic），即在每个状态下根据一定的概率分布选择动作。**

状态价值

状态价值函数（State-Value Function）表示在给定状态下，按照某种策略 π 进行决策所能获得的回报期望值，用 $V_\pi(s)$ 表示，如式 (11) 所示。

$$\begin{aligned} V_\pi(s) &= \mathbb{E}_\pi[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots | S_t = s] \\ &= \mathbb{E}_\pi[G_t | S_t = s] \end{aligned} \tag{11}$$

举例来说，假设智能体处于一个 3×3 的网格世界中，目标是走到右下角，对应获得 10 分的奖励，走到其他格子的奖励为 0，策略是随机选择一个方向（上、下、左、右）移动一步。

如果初始状态 s_0 即起点在左上角，按照当前策略，平均可能需要 10 才能到达终点，每步奖励为 0，到达终点时获得 10 分奖励，设折扣因子 $\gamma = 0.9$，则从起点状态出发的状态价值计算如式 (12) 所示。

$$V_\pi(s_0) = 0 + 0.9^1 \times 0 + 0.9^2 \times 0 + \cdots + 0.9^{10} \times 10 \approx 3.49 \tag{12}$$

动作价值

动作价值函数（Action-Value Function）表示在给定状态 s 和动作 a 下，按照某种策略 π 进行决策所能获得的回报期望值，用 $Q_\pi(s, a)$ 表示，如式 (13) 所示。

$$Q_\pi(s, a) = \mathbb{E}_\pi \left[G_t \mid s_t = s, a_t = a \right] \tag{13}$$

状态价值与动作价值的关系

状态价值函数和动作价值函数之间存在密切的关系，如式 (14) 所示。

$$V_\pi(s) = \sum_{a \in A} \pi(a \mid s) Q_\pi(s, a) \tag{14}$$

换句话说，状态价值是对所有可能的动作价值的加权平均，而动作价值则是对特定动作的评价。

状态价值反映了策略本身的好坏，它不关心智能体在状态 s 下选择了哪个具体动作，而是关注在该状态下按照策略 π 进行决策所能获得的整体回报期望值。动作价值则更具体地反映了在特定状态下选择某个动作所能获得的回报期望值，即不仅考虑智能体所处的状态，还考虑了智能体在该状态下选择的具体动作。

有模型与无模型

在谈到有模型（Model-Based）与无模型（Model-Free）方法时，这里的模型通常指与智能体交互的环境模型，即对环境的状态转移概率和奖励函数进行建模。根据是否使用环境模型，强化学习方法可以分为有模型方法和无模型方法两大类。

有模型的方法利用环境模型来进行规划和决策，通常包括动态规划（Dynamic Programming）等方法。这些方法通过对环境的状态转移概率和奖励函数进行建模，能够在不与环境直接交互的情况下，预测未来的状态和奖励，从而制定最优策略。

无模型的方法则不依赖于环境模型，包括 Q-Learning 、SARSA 等算法，这些方法通过与环境的直接交互，学习状态价值函数或动作价值函数，从而逐步改进策略。无模型方法通常更适用于复杂或未知的环境，因为它们不需要对环境进行显式建模，在强化学习中应用更为广泛。

预测与控制

预测（Prediction）问题是指在给定策略 π 的情况下，评估该策略的好坏，即计算状态价值函数 $V_\pi(s)$ 或动作价值函数 $Q_\pi(s,a)$。预测问题的目标是了解在当前策略下，智能体在不同状态下能够获得的回报期望值。用于预测任务的算法主要包括蒙特卡洛方法（Monte Carlo）、时序差分学习（Temporal Difference, TD）等，后续会详细介绍。

控制（Control）问题是指在给定环境模型的情况下，寻找最优策略 π^*，使得在每个状态下选择的动作能够最大化累积的回报。控制问题的目标是通过与环境的交互，学习一个最优策略，使得智能体能够在不同状态下获得最大的回报。用于控制任务的算法主要包括动态规划（Dynamic Programming）、Q学习（Q-Learning）、策略梯度方法（Policy Gradient）等，后续也会详细介绍。

复杂问题中通常需要同时解决预测和控制问题，即在学习最优策略的过程中，同时评估当前策略的好坏，从而不断改进策略，最终收敛到最优策略。

思考

强化学习所解决的问题一定要严格满足马尔可夫性质吗？请举例说明。

不一定。例如在围棋游戏场景中，不仅需要考虑当前棋子的位置，还需要考虑棋子的历史位置，因此不满足马尔可夫性质。但依然可以使用强化学习的方法进行求解，例如在 AlphaGO 论文中使用了蒙特卡洛树搜索算法来解决这个问题。在一些时序性场景中，也可以通过引入记忆单元来解决这个问题，例如在 DQN 算法中，使用了记忆单元来存储历史状态，从而解决了这个问题，尽管它也不满足马尔可夫性质。

马尔可夫决策过程主要包含哪些要素?

马尔可夫决策 $< S, A, R, P, \gamma >$ 主要包含状态空间 S、动作空间 A、奖励函数 R、状态转移矩阵 P、折扣因子 γ 等要素，其中状态转移矩阵 P 是环境的一部分，而其他要素是智能体的一部分。在实际应用中，通常还考虑值函数 V 和策略函数 π 等要素，值函数用于某个状态下的长期累积奖励，策略函数用于某个状态下的动作选择。

马尔可夫决策过程引入折扣因子 γ 的作用是什么？如何理解折扣因子的意义?

折扣因子 γ 的作用是用来控制未来奖励在当前决策中的重要性。它的取值范围在 0 到 1 之间。当 γ 接近 0 时，智能体更关注当前的奖励，而忽略未来的奖励；当 γ 接近 1 时，智能体会更加重视未来的奖励。一方面在数学上能够确保回报 G_t 的收敛性，另一方面也代表了时间价值，就像经济学中的货币折现，同样面值的货币现在拿到手中比未来拿到更有价值。

马尔可夫决策过程与金融科学中的马尔可夫链有什么区别与联系?

马尔可夫链是一个随机过程，其下一个状态只依赖于当前状态而不受历史状态的影响，即满足马尔可夫性质。马尔可夫链由状态空间、初始状态分布和状态转移概率矩阵组成。马尔可夫决策过程是一种基于马尔可夫链的决策模型，它包含了状态、行动、转移概率、奖励、值函数和策略等要素。马尔可夫决策过程中的状态和状态转移概率满足马尔可夫性质，但区别在于它还包括了行动、奖励、值函数和策略等要素，用于描述在给定状态下代理如何选择行动以获得最大的长期奖励。

有模型与免模型算法的区别？举一些相关的算法?

有模型算法在学习过程中使用环境模型，即环境的转移函数和奖励函数，来推断出最优策略。这种算法会先学习环境模型，然后使用模型来生成策略。因此，有模型算法需要对环境进行建模，需要先了解环境的转移函数和奖励函数，例如动态规划等算法。免模型算法不需要环境模型，而是直接通过试错来学习最优策略。这种算法会通过与环境的交互来学习策略，不需要先了解环境的转移函数和奖励函数。免模型算法可以直接从经验中学习，因此更加灵活，例如Q-learning 、Sarsa 等算法。

举例说明预测与控制的区别与联系。

区别：预测任务主要是关注如何预测当前状态或动作的价值或概率分布等信息，而不涉及选择动作的问题；控制任务则是在预测的基础上，通过选择合适的动作来最大化累计奖励，即学习一个最优的策略。**联系**：预测任务是控制任务的基础，因为在控制任务中需要对当前状态或动作进行预测才能选择最优的动作；控制任务中的策略通常是根据预测任务中获得的状态或动作价值函数来得到的，因此预测任务对于学习最优策略是至关重要的。以赌博机问题为例，预测任务是估计每个赌博机的期望奖励（即价值函数），控制任务是选择最优的赌博机来最大化累计奖励。在预测任务中，我们可以使用多种算法来估计每个赌博机的期望奖励，如蒙特卡罗方法、时间差分方法等。在控制任务中，我们可以使用贪心策略或ε-贪心策略来选择赌博机，这些策略通常是根据预测任务中得到的每个赌博机的价值函数来确定的。因此，预测任务对于控制任务的实现至关重要。