**智能体(agent)**：程序操纵的对象，例如机器人、小车、游戏角色等。

**环境(environment)**：这是智能体所处的世界，例如真实的物理世界、游戏世界等。它包括解决所发生情况所需的所有机制，例如真实世界中的物理规律、游戏的规则。

**状态(state)**：状态是智能体对当前世界的观察，例如机器人通过传感器观察周围的世界，或是在游戏中的像素矩阵。需要指出的是，状态需要满足马尔可夫性，观察可能不足以描述一个状态。用S/s来表示。

**行为(action)**：智能体做出的会改变当前状态的动作。例如机器人位置移动、游戏角色发起攻击等。用A/a来表示。

**策略(policy)**：智能体在每一种状态下应该做出何种行为。例如机器人在遇到障碍物时应该避开、游戏角色遇见敌人应该发起进攻。用来表示，，含义是在某个状态下采取某种行为的概率。

**奖励(reward)**：智能体做出某种行为后，环境对其的反馈。例如机器人撞上障碍物，环境给一个很大的负奖励、游戏角色吃到金币，环境给一个正奖励。用R/r来表示。

智能体

环境

状态

奖励

行为

上图说明了智能体和环境之间的互动：环境将状态和奖励反馈给智能体，智能体根据这些反馈产生行为，行为会改变环境，而后环境将新的状态和奖励反馈给智能体，周而复始。其中，存在两种随机性，而这两种随机性在强化学习中非常关键。第一是某种状态下的智能体的某种行为，导致环境发生的变化具有随机性，记这种随机性为；第二种是智能体本身做决定的随机性，这种随机性由智能体本身的策略决定。例如有k个抽奖机，每个抽奖机都有一定的概率中奖，而智能体每次选择一个抽奖机抽奖，假设智能体选择的策略是：每次以的概率选择历史中奖率最高的抽奖机抽奖，的概率随机选择一个抽奖机抽奖。则每次智能体做出选择一个抽奖机抽奖这个行为后，是否中奖是随机的，等于第i台抽奖机的中奖概率；而选择哪一台抽奖机进行抽奖也是随机的。

**未来累计奖励(Cumulative future reward)**：表示在一系列确定的行为、状态下能获得的奖励。例如在石头剪刀布游戏中，采取一直出石头的策略，经过三局比赛，能获得的奖励。记其为，其中，为折扣因子(discount rate)，其意义是使得距当前更近的奖励有更大的权重。

**行为-价值函数(action-value function)**：其定义为在某种策略下，智能体在某个状态下，采取某种行为后，能获得的未来累计奖励的期望。例如在剪刀石头布游戏中，采取一直出石头的策略，若第一局出布，则其能获得的未来累计奖励的期望。记其为，其意义是衡量在某个策略、某种状态下，采用某种行为的优劣。表示最优策略。

**状态-价值函数(state-value function)**：其定义为采取某种策略，在某个状态下的未来累计奖励的期望。记其为，例如在剪刀石头布游戏中，采取80%出石头，10%出布和10%出剪刀的策略，则。状态-价值函数反映在当前状态下该策略的好坏，而状态-价值函数对于所有状态的期望反映当前策略的好坏。

**蒙特卡洛法(Monte Carlo algorithm)**：多次采样，用平均累计奖赏作为期望累计奖赏的近似。例如在和某个固定的对手下围棋过程中，采用某种策略，下多盘棋，可以得到这种策略的平均胜率，即可作为价值函数的估计值。这种方法的缺点就是需要完成整个过程，才能计算得到奖赏，比如需要把一盘棋下完才能得到累计奖赏的结果。

**时序差分算法(Temporal difference algorithm)**：利用强化学习的MDP结构，利用t+1步的价值函数来估计第t步的价值函数。推导如下(此处使用Q函数进行推导，V函数同理)：

其中，表示第t步真实的奖励。其中，称为**TD目标(TD target)**，记为。使用一个直觉性的例子解释该公式：某人开车从A经过B到达C，在A时，预估需要10小时到达C，经过3小时到达B后，预估还需要6小时到达C。此时，就可以使用3+6=9小时来估计从A到C的时间，这个9小时，就是TD目标，而TD目标和最开始估计的10小时之间的差距，就是**TD误差(TD Error)**，记为。

**Sarsa算法**：对于离散的情况，使用对算法进行更新，是学习率。

**Q-learning算法**：对于离散的情况，使用对算法进行更新，其与Sarsa算法的区别是其只对最优的价值函数进行更新。对于连续的情况使用DQN等方法进行近似（DQN在下一部分介绍）。

**价值学习(value-based method)**：通过更新价值函数来进行学习的方法，此处以DQN(deep Q-learning)为例。DQN使用一个神经网络来近似行为-价值函数，即，其中w是网络的参数，s是输入的状态，例如在游戏中s是当前画面的像素矩阵，Q的输出是智能体在该种状态下做出每种行为的概率。之后DQN算法使用梯度下降法对网络的参数进行更新，损失函数写为：

其中，是TD目标，故有：

**策略学习(policy-based method)**：通过更新策略来进行学习，这样同样是介绍基于神经网络的策略学习算法，同样使用一个神经网络来状态-价值函数：，使用各个状态下的平均状态-价值函数作为损失函数：，该策略函数反映该损失函数的好坏；根据，有

其中，可以使用蒙特卡洛法进行估计获得。

**Actor-Critic方法**：用一个神经网络来近似价值函数，该网络称为critic，就像裁判员；用另一个神经网络来近似策略，该网络称为actor，就像跳水运动员。Actor和Critic的关系如下图所示：

策略网络(critic)

价值网络(actor)

环境

状态

行为

奖励

值