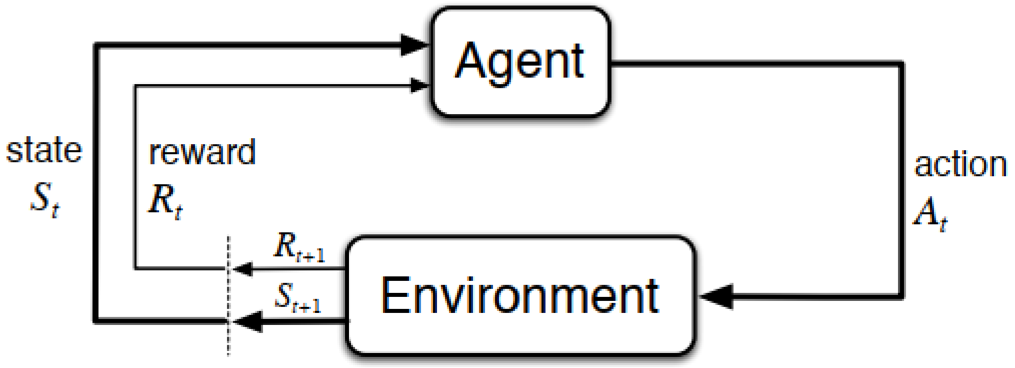
Markov Decision Processes

智能体和环境

概念

- The learner and decision maker is called the *agent*.
- The thing it interacts with, comprising everything outside the agent, is called the *environment*. 与智能体交互的事情，构成了其外部的所有
- These interact continually, the agent selecting actions and the environment responding to these actions and presenting new situations to the agent.
- The environment also gives rise to rewards, special numerical values that the agent seeks to maximize over time through its choice of actions.
- 

- learning to control a system so as to maximize some numerical value which represents a long-term objective

界限

- anything that cannot be changed arbitrarily by the agent is considered to be outside of it and thus part of its environment
    - 机器人的传感器属于外部环境，人的骨骼、肌肉属于外部环境
    - 对于智能体来说环境中的事物不都是未知的，即使知道了所有环境，面对的问题可能也很难（魔方）
    - 智能体不能任意改变的事务就是它外部的
- The agent–environment boundary represents the limit of the agent's *absolute control*, not of its knowledge.
- Everything inside the agent is completely known and controllable by the agent.
- Everything outside is incompletely controllable but may or may not be completely known.

交互过程

概念

- The agent and environment interact at each of a sequence of discrete time steps. 智能体和环境的交互是一个离散的时间序列
- At each time step t, the agent receives some representation of the environment's state, $S_t \in \mathcal{S}$, and on that basis selects an action, $A_t \in \mathcal{A}(s)$. 每个时间点，智能体得到环境的一个表示，基于此选择一个动作
- One time step later, in part as a consequence of its action, the agent receives a numerical reward, $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$, and finds itself in a new state, S_{t+1}. 到达下一个时间点，由于上一个时间点的动作，智能体得到一个数值回报，并且到达新状态

Episodes

episodic tasks

- when the agent–environment interaction breaks naturally into subsequences, which we call *episodes*

  在交互过程中自然停止，称为一段 episode 或者 trial
- Each episode ends in a special state called the *terminal state*, followed by a reset to a starting state.
- the next episode begins independently of how the previous one ended 序列间相互独立

continuing tasks

- in many cases the agent–environment interaction does not break naturally into identifiable episodes, but goes on continually without limit
- *absorbing state* that transitions only to itself and that generates only rewards of zero

MDP 框架

框架

- In general, actions can be any decisions we want to learn how to make, and the states can be anything we can know that might be useful in making them.
- The MDP framework is a considerable abstraction of the problem of goal-directed learning from interaction.

  目的导向的问题抽象
- Any problem of learning goal-directed behavior can be reduced to three signals passing back and forth between an agent and its environment: one signal to represent the choices made by the agent (the actions), one signal to represent the basis on which the choices are made (the states), and one signal to define the agent's goal (the rewards).

  所有目的导向的问题都可以转化为MDP（选择，选择的基础，目标）
- 最佳效果的状态和动作的表示，暂时没有一种科学的方法去构建

  such representational choices are at present more art than science
- Markov Decision Processes are a tool for modeling sequential decision-making problems where a decision maker interacts with a system in a sequential fashion. 适用于序列性的决策问题
- $(\mathcal{S}; \mathcal{A}; \mathcal{P}; \mathcal{R}; \gamma)$ 定义在五元组上
- In problems of *complete knowledge*, the agent has a complete and accurate model of the environment's dynamics.

  In problems of *incomplete knowledge*, a complete and perfect model of the environment is not available.

见微知著

- time step 不一定是固定时长的真实时间，可以是一系列连续的阶段

  The time steps need not refer to fixed intervals of real time; they can refer to arbitrary successive stages of decision making and acting.
- 动作可大可小，可宏观可微观
  - The actions can be low-level controls, such as the voltages applied to the motors of a robot arm, or high-level decisions, such as whether or not to have lunch or to go to graduate school.
  - some actions might be totally mental or computational
  - some actions might control what an agent chooses to think about, or where it focuses its attention
  - A_t is the action prescribed by the behavior based on the history $X_0, A_0, R_1; \ldots; X_{t-1}, A_{t-1}, R_t; X_t$.
- 状态也具有多种形式，可主观可客观
  - They can be completely determined by low-level sensations, such as direct sensor readings, or they can be more high-level and abstract, such as symbolic descriptions of objects in a room.
  - Some of what makes up a state could be based on memory of past sensations or even be entirely mental or subjective.
  - An agent could be in the state of not being sure where an object is, or of having just been surprised in some clearly defined sense.
  - The state must include information about all aspects of the past agent–environment interaction that make a difference for the future. If it does, then the state is said to have the *Markov property*.

状态包含了所有对未来起作用的历史交互信息

- ## Rewards

  - the purpose or goal of the agent is formalized in terms of a special signal, called the *reward*, passing from the environment to the agent
  - It is thus critical that the rewards we set up truly indicate what we want accomplished.
  - The reward signal is your way of communicating to the robot what you want it to achieve, not how you want it achieved.

    设置回报的目的是告诉智能体我想达到什么目标，而不是怎么样达到

- ## Returns

  - the agent’s goal is to maximize the cumulative reward it receives in the long run

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

    - γ is a parameter, $0 \leq \gamma \leq 1$, called the *discount rate*.
    - If $\gamma < 1$, the infinite sum in has a finite value (*discounted reward*)
    - If $\gamma = 0$, the agent is “myopic” in being concerned only with maximizing immediate rewards
    - As γ approaches 1, the return objective takes future rewards into account more strongly; the agent becomes more farsighted. In an episodic MDP, we often consider undiscounted rewards.
    - $\gamma = 1$ *undiscounted*
  - 马上得到的回馈比未来得到的更有价值，越往后，不确定性越大
    - γ 的设置可能并不合理，违背了最大化 reward 的初衷，减小靠后时间的权重，不一视同仁，感觉说不通。
    - 个人的想法，K 步为一组，reward 设一个最小阈值，若 K 步内的最大值超过阈值，继续看后面 K 步，循环往复；若 K 步内的最大值没有超过阈值，return 的计算就到此为止；若超过 nK 步，依然最大值超过阈值，也停下来不再往后看
    - K 步内都没超过阈值，认为这个序列并不优秀，后面的状态也就不考虑了；nK 步了，依然超过阈值，认为序列还不错，还可以在此基础上加一个奖励
  - $$\begin{aligned} G_t &\doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots \\ &= R_{t+1} + \gamma \left(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \cdots\right) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned}$$
  - 目标是找到选取动作的方式使得累计回报最大 The goal of the decision maker is to come up with a way of choosing the actions so as to maximize the expected total discounted reward, irrespectively of how the process is started. Such a maximizing behavior is said to be *optimal*.
  - undiscounted formulation is appropriate for episodic tasks

    discounted formulation is appropriate for continuing tasks

- ## 无后效性

  - In a finite MDP, the sets of states, actions, and rewards $(\mathcal{S}, \mathcal{A}, \text{ and } \mathcal{R})$ all have a finite number of elements. In this case, the random variables R_t and S_t have well defined discrete probability distributions dependent only on the preceding state and action. 对于有限MDP，针对前一个动作和状态，它转移到随机的状态和得到随机的回报值，都对应一个离散的概率
  - for particular values of these random variables, $s' \in \mathcal{S}$ and $r \in \mathcal{R}$, there is a probability of those values occurring at time t, given particular values of the preceding state and action 只依赖于前一个状态和动作

    $$p\left(s', r | s, a\right) \doteq \operatorname{Pr}\{S_t = s', R_t = r | S_{t-1} = s, A_{t-1} = a\}$$

  - The dynamics function $p : S \times R \times S \times A \rightarrow [0,1]$ is an ordinary deterministic function of four arguments. 具有四个参数的确定函数，值域为[0, 1]，可以计算出所有关于环境的信息 代表了 **environment’s dynamics**
  - 条件概率

    $$\sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p\left(s', r | s, a\right) = 1, \text{ for all } s \in \mathcal{S}, a \in \mathcal{A}(s)$$

  - In a Markov decision process, the probabilities given by p completely characterize the environment’s dynamics.

    该概率函数完全刻画了环境的动态性

- the probability of each possible value for S_t and R_t depends only on the immediately preceding state S_{t-1} and action A_{t-1}

  限制不在于决策过程，而是状态

概率函数

- *state-transition probabilities 状态转移概率*

$$p\left(s'|s,a\right) \doteq \Pr\{S_t = s'|S_{t-1} = s, A_{t-1} = a\} = \sum_{r\in\mathcal{R}} p\left(s', r|s, a\right)$$
$$p: \mathcal{S}\times\mathcal{S}\times\mathcal{A}\rightarrow[0,1])$$

- expected rewards for state-action pairs *立即回报函数 immediate reward function*

$$r(s,a) \doteq \mathbb{E}\left[R_t|S_{t-1} = s, A_{t-1} = a\right] = \sum_{r\in\mathcal{R}} r \sum_{s'\in\mathcal{S}} p\left(s', r|s, a\right)$$
$$r: \mathcal{S}\times\mathcal{A}\rightarrow\mathbb{R}$$

- expected rewards for state-action-next state triples

$$r\left(s,a,s'\right) \doteq \mathbb{E}\left[R_t|S_{t-1} = s, A_{t-1} = a, S_t = s'\right] = \sum_{r\in\mathcal{R}} r\frac{p\left(s', r|s, a\right)}{p\left(s'|s,a\right)}$$
$$r: \mathcal{S}\times\mathcal{A}\times\mathcal{S}\rightarrow\mathbb{R}$$

Policy

- particular ways of acting, called policies (A rule describing the way the actions are selected)
- a *policy* is a mapping from states to probabilities of selecting each possible action.
- *Deterministic stationary policies*，状态到动作的映射；*stochastic stationary policy*，状态到动作空间的分布的映射
- If the agent is following policy π at time t, then $\pi(a|s)$ is the probability that $A_t = a$ if $S_t = s$.

  defines a probability distribution over $a\in\mathcal{A}(s)$ for each $s\in\mathcal{S}$

Value Functions

- functions of states (or of state–action pairs) that estimate how good it is for the agent to be in a given state (or how good it is to perform a given action in a given state)
- A value function is a prediction of the expected, accumulative, discounted, future reward, measuring how good each state, or state-action pair.
- The value function of a state s under a policy π, denoted $v_\pi(s)$, *state-value function for policy π*

  The value of taking action a in state s under a policy π, denoted $q_\pi(s,a)$, *action-value function for policy π*

$$v_\pi(s) \doteq \mathbb{E}_\pi\left[G_t|S_t = s\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t = s\right], \quad \forall s\in\mathcal{S}$$
$$q_\pi(s,a) \doteq \mathbb{E}_\pi\left[G_t|S_t = s, A_t = a\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t = s, A_t = a\right]$$
$$v_\pi(s) = \sum_{a\in\mathcal{A}}\pi(a|s)q_\pi(s,a)$$
$$q_\pi(s,a) = \sum_{s',r} p\left(s', r|s, a\right)\left[r + \gamma v_\pi\left(s'\right)\right]$$

贝尔曼方程 Bellman optimality equation

Value Functions with Successor States

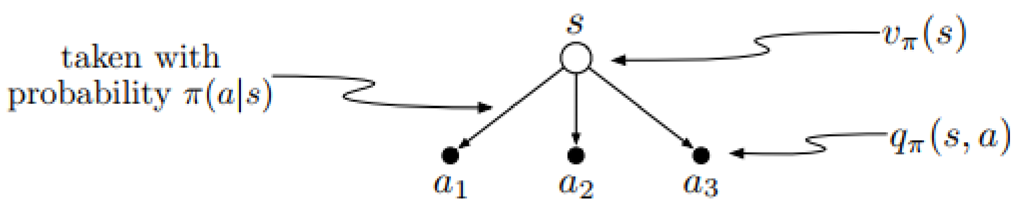
$$\begin{aligned} v_\pi(s) &\doteq \mathbb{E}_\pi\left[G_t \mid S_t=s\right] \\ &=\mathbb{E}_\pi\left[R_{t+1}+\gamma G_{t+1} \mid S_t=s\right] \\ &=\sum_a \pi(a \mid s) \sum_{s^{\prime}} \sum_r p\left(s^{\prime}, r \mid s, a\right)\left[r+\gamma \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1}=s^{\prime}\right]\right] \\ &=\sum_a \pi(a \mid s) \sum_{s^{\prime}, r} p\left(s^{\prime}, r \mid s, a\right)\left[r+\gamma v_\pi\left(s^{\prime}\right)\right], \quad \text { for all } s \in \mathcal{S} \end{aligned}$$

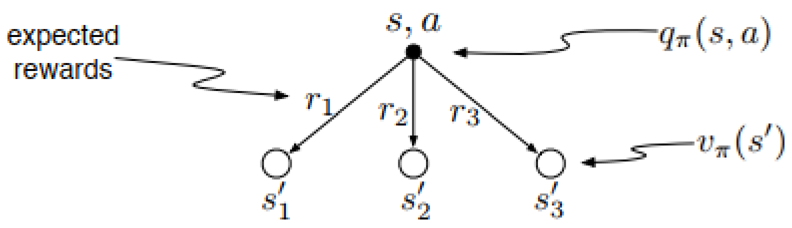
$$q_\pi(s, a)=\sum_{s^{\prime}, r} p\left(s^{\prime}, r \mid s, a\right)\left[r+\gamma v_\pi(s^{\prime})\right]=\sum_{s^{\prime}, r} p\left(s^{\prime}, r \mid s, a\right)\left[r+\gamma \sum_{a^{\prime}} \pi\left(a^{\prime} \mid s^{\prime}\right) q_\pi\left(s^{\prime}, a^{\prime}\right)\right]$$

$$\begin{aligned} v(s) &= \mathbb{E}\left[R_{t+1}+\gamma v\left(S_{t+1}\right) \mid S_t=s\right] \\ q(s, a) &= \mathbb{E}\left[R_{t+1}+\gamma q\left(S_{t+1}, A_{t+1}\right) \mid S_t=s, A_t=a\right] \end{aligned}$$

- reward 和 return **都是基于某个状态而言的，都是期望值**（G_t 和 $v(s)$ or $q(s,a)$ 可以看成一体两面）

图解





Optimal Policies and Optimal Value Functions

贪心算法

$$v_*(s) \doteq \max_\pi v_\pi(s)$$

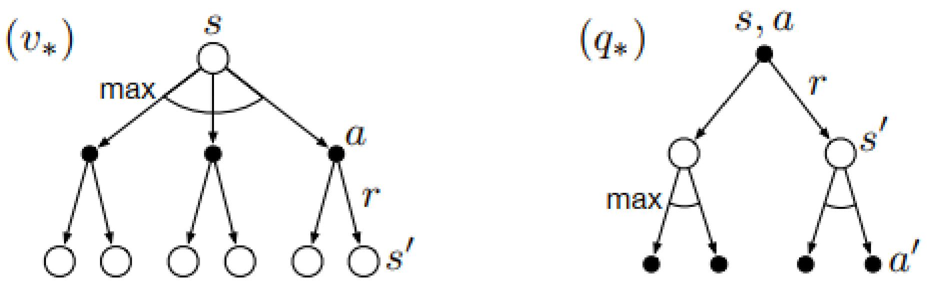
$$\begin{aligned} v_*(s) &= \max_{a \in \mathcal{A}(s)} q_*(s, a) \\ &=\max_a \mathbb{E}\left[G_t \mid S_t=s, A_t=a\right] \\ &=\max_a \mathbb{E}\left[R_{t+1}+\gamma G_{t+1} \mid S_t=s, A_t=a\right] \\ &=\max_a \mathbb{E}\left[R_{t+1}+\gamma v_*\left(S_{t+1}\right) \mid S_t=s, A_t=a\right] \\ &=\max_a \sum_{s^{\prime}, r} p\left(s^{\prime}, r \mid s, a\right)\left[r+\gamma v_*\left(s^{\prime}\right)\right] \end{aligned}$$

$$q_*(s, a) \doteq \max_\pi q_\pi(s, a)$$

$$\begin{aligned} q_*(s, a) &= \mathbb{E}\left[R_{t+1}+\gamma \max_{a^{\prime}} q_*\left(S_{t+1}, a^{\prime}\right) \mid S_t=s, A_t=a\right] \\ &=\sum_{s^{\prime}, r} p\left(s^{\prime}, r \mid s, a\right)\left[r+\gamma \max_{a^{\prime}} q_*\left(s^{\prime}, a^{\prime}\right)\right] \end{aligned}$$

$$q_*(s, a)=\mathbb{E}\left[R_{t+1}+\gamma v_*\left(S_{t+1}\right) \mid S_t=s, A_t=a\right]$$

图解



- 每一个状态，选择 return 最大的一个动作 (在某个范围中选max)
- 每一个状态-动作，对可能到达的状态求和，下一个状态会选择 return 最大的动作

  taking action a in state s and thereafter following an optimal policy

- the value of a state under an optimal policy must equal the expected return for the best action from that state

可解性

- 对于 $v_*(s)$，n 个 states，n 个变量，n 个等式，p 已知的情况下可解
- For each state s, there will be one or more actions at which the maximum is obtained in the Bellman optimality equation. Any policy that assigns nonzero probability only to these actions is an optimal policy.
- $v_*(s)$ 已经考虑到了未来的行为，贪心即是最优的解法

Optimality and Approximation

- optimal policies can be generated only with extreme computational cost
- memory available is also an important constraint (there are far more states than could possibly be entries in a table)
- in approximating optimal behavior, there may be many states that the agent faces with such a low probability that selecting suboptimal actions for them has little impact on the amount of reward the agent receives
- put more effort into learning to make good decisions for frequently encountered states, at the expense of less effort for infrequently encountered states
- 尽量优化出现概率大的状态；对于低概率的状态，不能选择最优策略也可以接受