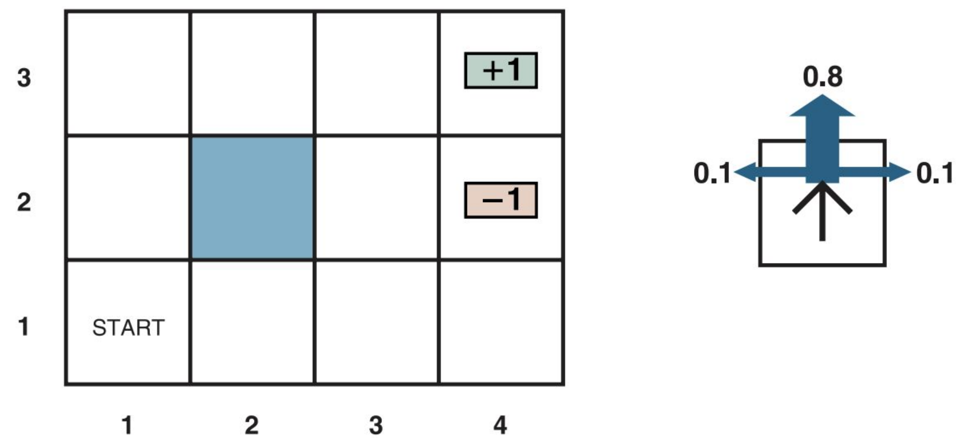
Markov Chains

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} a_{12} \ldots a_{n1} \ldots a_{nn}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$\pi = \pi_1, \pi_2, \ldots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

Hidden Markov Models

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} \ldots a_{ij} \ldots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \ldots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, ..., v_V$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state i
$\pi = \pi_1, \pi_2, ..., \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

A Markov Decision Problem
(source: Russell & Norvig)



(a) (b)

Figure 17.1 (a) A simple, stochastic 4×3 environment that presents the agent with a sequential decision problem. (b) Illustration of the transition model of the environment: the "intended" outcome occurs with probability 0.8, but with probability 0.2 the agent moves at right angles to the intended direction. A collision with a wall results in no movement. Transitions into the two terminal states have reward +1 and –1, respectively, and all other transitions have a reward of –0.04.

The optimal policy

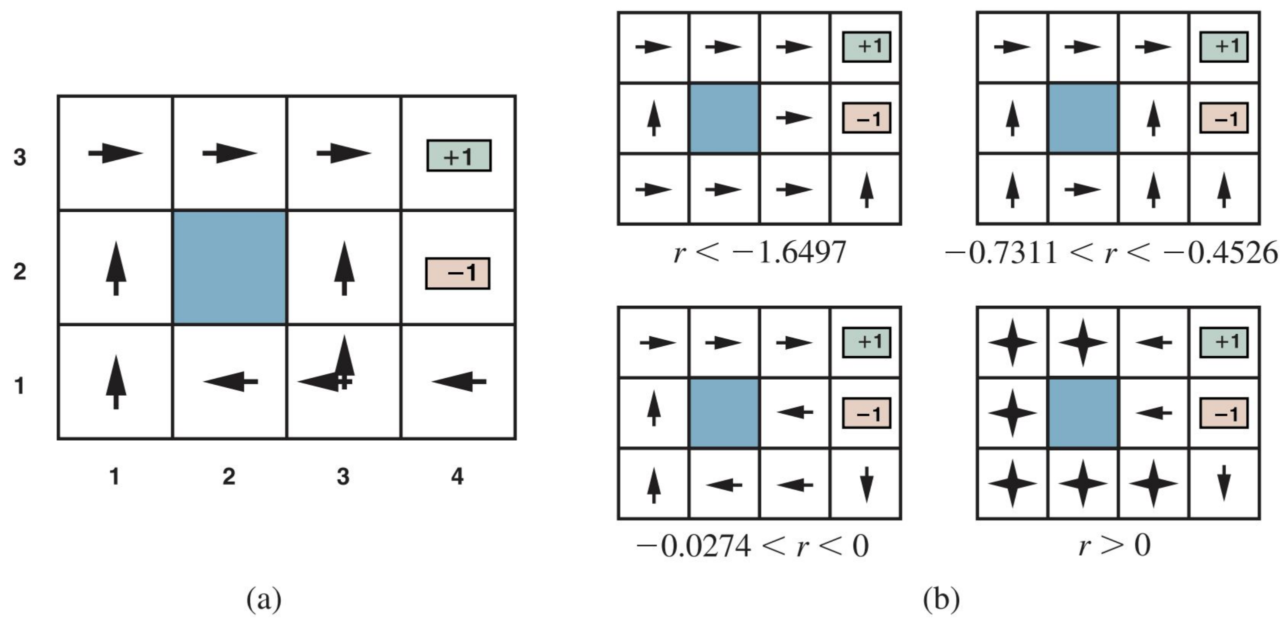


Figure 17.2 (a) The optimal policies for the stochastic environment with $r = -0.04$ for transitions between nonterminal states. There are two policies because in state (3,1) both *Left* and *Up* are optimal. (b) Optimal policies for four different ranges of r.

The value function corresponding to the optimal policy

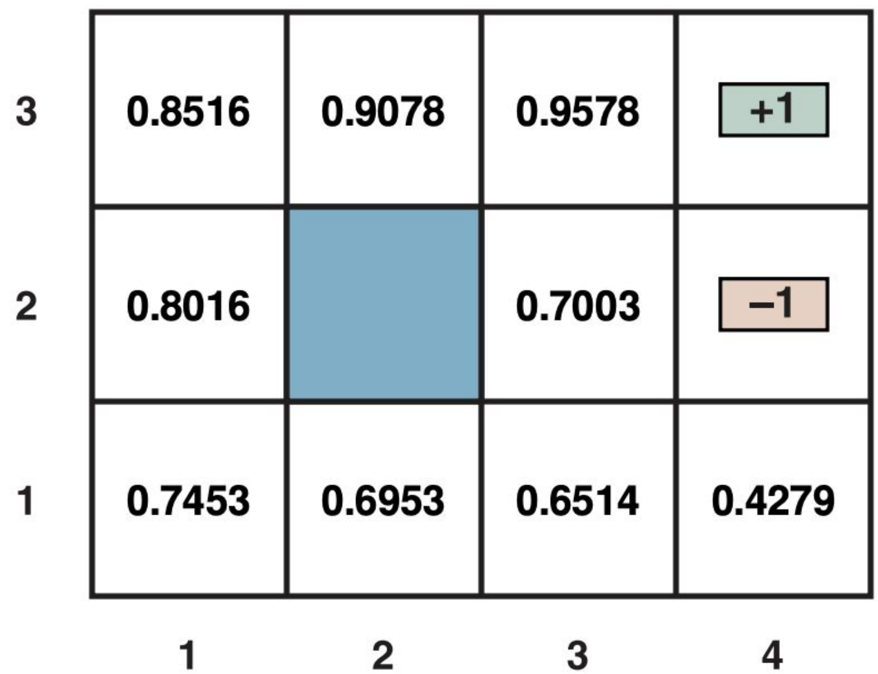


Figure 17.3 The utilities of the states in the 4×3 world with $\gamma = 1$ and $r = -0.04$ for transitions to nonterminal states.

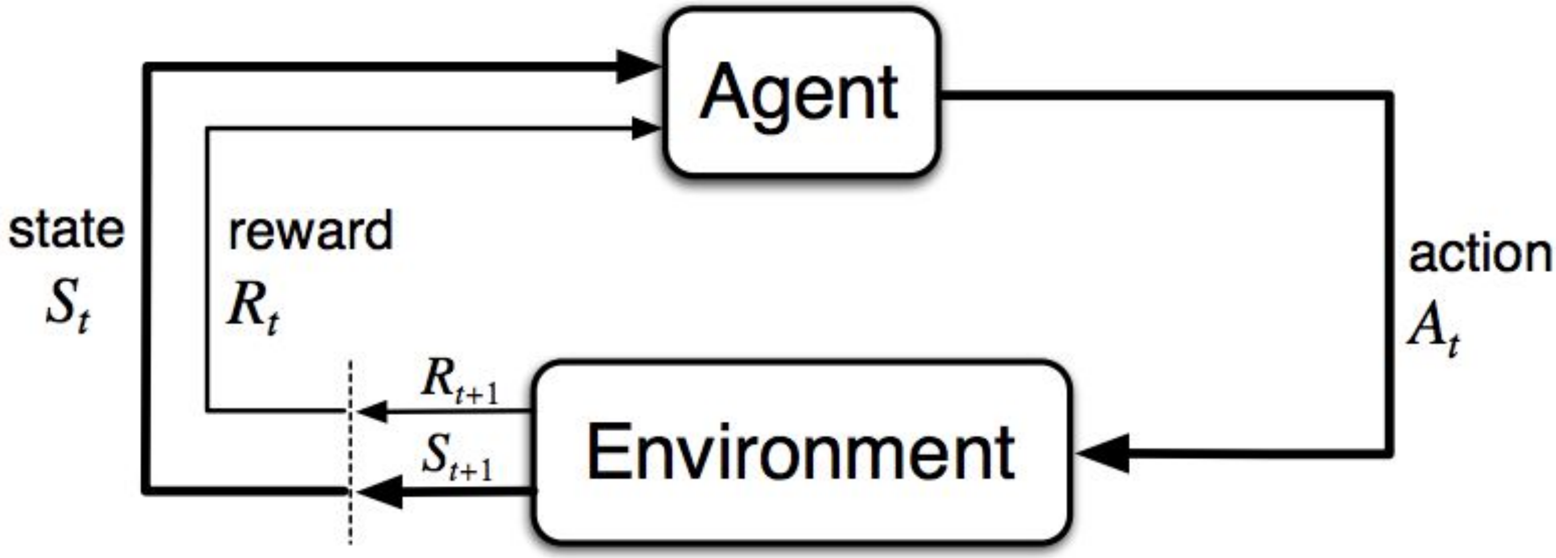


Figure 3.1: The agent–environment interaction in reinforcement learning.

Source: Sutton and Barto

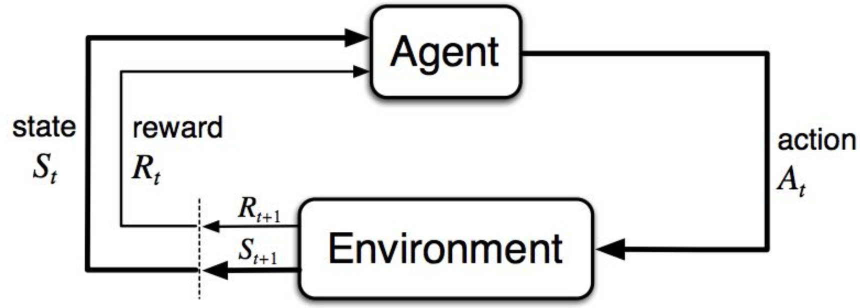


Figure 3.1: The agent–environment interaction in reinforcement learning.

More specifically, the agent and environment interact at each of a sequence of discrete time steps, $t = 0, 1, 2, 3, \ldots$.[2] At each time step t, the agent receives some representation of the environment's *state*, $S_t \in \mathcal{S}$, where $\mathcal{S}$ is the set of possible states, and on that basis selects an *action*, $A_t \in \mathcal{A}(S_t)$, where $\mathcal{A}(S_t)$ is the set of actions available in state S_t. 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}.[3] Figure 3.1 diagrams the agent–environment interaction.

At each time step, the agent implements a mapping from states to probabilities of selecting each possible action. This mapping is called the agent's *policy* and is denoted π_t, where $\pi_t(a|s)$ is the probability that $A_t = a$ if $S_t = s$.

Some examples

- Playing a game like Chess, Go, Backgammon etc.
- Managing your stock portfolio
- Learning to walk
- Managing the inventory at a car-rental company with multiple locations

The Markov property

Consider how a general environment might respond at time $t+1$ to the action taken at time t. In the most general, causal case, this response may depend on everything that has happened earlier. In this case the dynamics can be defined only by specifying the complete joint probability distribution:

$$\Pr\{S_{t+1} = s', R_{t+1} = r \mid S_0, A_0, R_1, \ldots, S_{t-1}, A_{t-1}, R_t, S_t, A_t\}, \tag{3.4}$$

for all r, s', and all possible values of the past events: S_0, A_0, R_1, ..., S_{t-1}, A_{t-1}, R_t, S_t, A_t. If the state signal has the *Markov property*, on the other hand, then the environment's response at $t+1$ depends only on the state and action representations at t, in which case the environment's dynamics can be defined by specifying only

$$p(s', r|s, a) \doteq \Pr\{S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a\}, \tag{3.5}$$

for all r, s', s, and a. In other words, a state signal has the Markov property, and is a Markov state, if and only if (3.4) is equal to $p(s', r|S_t, A_t)$ for all s', r, and histories, S_0, A_0, R_1, ..., S_{t-1}, A_{t-1}, R_t, S_t, A_t. In this case, the environment and task as a whole are also said to have the Markov property.

Markov Decision Processes

A particular finite MDP is defined by its state and action sets and by the one-step dynamics of the environment. Given any state and action s and a, the probability of each possible pair of next state and reward, s', r, is denoted

$$p(s', r|s, a) = \Pr\{S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a\}. \tag{3.6}$$

Value function and Q function

Recall that a policy, π, is a mapping from each state, $s \in \mathcal{S}$, and action, $a \in \mathcal{A}(s)$, to the probability $\pi(a|s)$ of taking action a when in state s. Informally, the *value* of a state s under a policy π, denoted $v_\pi(s)$, is the expected return when starting in s and following π thereafter. For MDPs, we can define $v_\pi(s)$ formally as

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right], \tag{3.10}$$

where $\mathbb{E}_\pi[\cdot]$ denotes the expected value of a random variable given that the agent follows policy π, and t is any time step. Note that the value of the terminal state, if any, is always zero. We call the function v_π the *state-value function for policy* π.

Similarly, we define the value of taking action a in state s under a policy π, denoted $q_\pi(s,a)$, as the expected return starting from s, taking the action a, and thereafter following policy π:

$$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a\right]. \tag{3.11}$$

We call q_π the *action-value function for policy* π.

Estimating V_π(s) and q_π(s, a) from experience

- If an agent follows policy π and maintains an average, for each state encountered, of the actual returns that have followed that state, then the average will converge to the state's value V_π (s)
- Similarly, if the agent performs action a in state s, and thereafter follows policy π, the average of the returns will converge to q_π(s,a)
- If we don't know the transition probabilities explicitly, but have access to a simulator that is good enough!

The Bellman optimality equation for V_{π}(s)

$$
\begin{aligned}
v_*(s) &= \max_{a\in\mathcal{A}(s)} q_{\pi_*}(s,a) \\
&= \max_a \mathbb{E}_{\pi^*}[G_t \mid S_t = s, A_t = a] \\
&= \max_a \mathbb{E}_{\pi^*}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a\right] \\
&= \max_a \mathbb{E}_{\pi^*}\left[R_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k R_{t+k+2} \,\middle|\, S_t = s, A_t = a\right] \\
&= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a] && (3.16) \\
&= \max_{a\in\mathcal{A}(s)} \sum_{s',r} p(s',r|s,a)\big[r + \gamma v_*(s')\big]. && (3.17)
\end{aligned}
$$

Value iteration

Initialize array V arbitrarily (e.g., $V(s) = 0$ for all $s \in \mathcal{S}^+$)

Repeat
  $\Delta \leftarrow 0$
  For each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)

Output a deterministic policy, $\pi \approx \pi_*$, such that
  $\pi(s) = \arg\max_a \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$

Policy iteration (using iterative policy evaluation)

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Repeat
   $\quad \Delta \leftarrow 0$
   $\quad$ For each $s \in \mathcal{S}$:
   $\quad\quad v \leftarrow V(s)$
   $\quad\quad V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))\big[r + \gamma V(s')\big]$
   $\quad\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $\quad$ *old-action* $\leftarrow \pi(s)$
   $\quad \pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$
   $\quad$ If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Deep Reinforcement Learning

- In most practical problems there are too many states to be listed explicitly. So we need to rely on function approximation methods. The policy / value functions are represented by neural networks. This is called Deep RL.
- There are many ways of learning from past experience. Roughly speaking you want to do more of the action sequences which give you high reward, and less of the action sequences which give you low reward. Hence the term “reinforcement” which originated in the animal learning literature
- One class of methods for training are “policy gradient” methods, of which a leading example is PPO.
- I recommend the full series on Deep RL hosted by Pieter Abbeel

  https://www.youtube.com/watch?v=qaMdN6LS9rA
- https://www.youtube.com/watch?v=tqrcjHuNdmQ is a pretty intuitive introduction to policy gradients from Karpathy