# 3 Finite Markov Decision Processes

We say that a system has the *Markov property* if each state includes all information about the previous states and actions that makes a difference to the future.

The MDP provides an abstraction of the problem of goal-directed learning from interaction by modelling the whole thing as three signals: action, state, reward.

Together, the MDP and agent give rise to the *trajectory* $S_0$, $A_0$, $R_1$, $S_1$, $A_1$, $S_2$, $R_2$, $\ldots$. The action choice in a state gives rise (stochastically) to a state and corresponding reward.

## 3.1 The Agent–Environment Interface

We consider finite Markov Decision Processes (MDPs). The word finite refers to the fact that the states, rewards and actions form a finite set. This framework is useful for many reinforcement learning problems.

We call the learner or decision making component of a system the *agent*. Everything else is the *environment*. General rule is that anything that the agent does not have absolute control over forms part of the environment. For a robot the environment would include it's physical machinery. The boundary is the limit of absolute control of the agent, not of its knowledge.

The MDP formulation is as follows. Index time-steps by $t \in \mathbb{N}$. Then actions, rewards, states at $t$ represented by $A_t \in \mathcal{A}(s)$, $R_t \in \mathcal{R} \subset \mathbb{R}$, $S_t \in \mathcal{S}$. Note that the set of available actions is dependent on the current state.

A key quantity in an MDP is the following function, which defines the *dynamics* of the system.

$$p(s', r|s, a) \doteq \mathbb{P}(S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a) \tag{8}$$

From this quantity we can get other useful functions. In particular we have the following:

**state-transition probabilities**

$$p(s'|s, a) \doteq \mathbb{P}(S_t = s'|S_{t-1} = s, A_{t-1} = A) = \sum_{r \in \mathcal{R}} p(s', r|s, a) \tag{9}$$

note the abuse of notation using $p$ again; and,

**expected reward**

$$r(s, a) = \mathbb{E}[R_t|S_{t-1} = s, A_{t-1} = a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r|s, a). \tag{10}$$

## 3.2 Goals and rewards

We have the *reward hypothesis*, which is a central assumption in reinforcement learning:

> All of what we mean by goals and purposes can be well thought of as the maximisation of the expected value of the cumulative sum of a received scalar signal (called reward).

## 3.3 Returns and Episodes

Denote the sequence of rewards from time $t$ as $R_{t+1}$, $R_{t+2}$, $R_{t+3}$, .... We seek to maximise the *expected return* $G_t$ which is some function of the rewards. The simplest case is where $G_t = \sum_{\tau > t} R_\tau$.

In some applications there is a natural final time-step which we denote $T$. The final time-step corresponds to a *terminal state* that breaks the agent-environment interaction into subsequences called *episodes*. Each episode ends in the same terminal state, possibly with a different reward. Each starts independently of the last, with some distribution of starting states. We denote the set of states including the terminal state as $\mathcal{S}^+$

Sequences of interaction without a terminal state are called *continuing tasks*.

We define $G_t$ using the notion of *discounting*, incorporating the *discount rate* $0 \leq \gamma \leq 1$. In this approach the agent chooses $A_t$ to maximise

$$G_t \doteq \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}. \tag{11}$$

This sum converges wherever the sequence $R_t$ is bounded. If $\gamma = 0$ the agent is said to be myopic. We define $G_T = 0$. Note that

$$G_t = R_{t+1} + \gamma G_{t+1}. \tag{12}$$

Note that in the case of finite time steps or an episodic problem, then the return for each episode is just the sum (or whatever function) of the returns in that episode.

## 3.4 Unified Notation for Episodic and Continuing Tasks

We want to unify the notation for episodic and continuing learning.

We introduce the concept of an *absorbing state*. This state transitions only to itself and gives reward of zero.

To incorporate the (disjoint) possibilites that $T = \infty$ or $\gamma = 1$ in our formulation of the return, we might like to write

$$G_t \doteq \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k. \tag{13}$$

## 3.5 Policies & Value Functions

### Policy

A *policy* $\pi(a|s)$ is a mapping from states to the probability of selecting actions in that state. If an agent is following policy $\pi$ and at time $t$ is in state $S_t$, then the probability of taking action $A_t$ is $\pi(a|s)$. Reinforcement learning is about altering the policy from experience.

### Value Functions

As we have seen, a central notion is the value of a state. The *state-value function* of state $s$ under policy $\pi$ is the expected return starting in $s$ and following $\pi$ thereafter. For MDPs this is

$$v_\pi \doteq \mathbb{E}_\pi[G_t | S_t = s], \tag{14}$$

where the subscript $\pi$ denotes that this is an expectation taken conditional on the agent following policy $\pi$.

Similarly, we define the *action-value function* for policy $\pi$ to be the expected return from taking action $a$ in state $s$ and following $\pi$ thereafter

$$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t|S_t = s, A_t = a]. \tag{15}$$

The value functions $v_\pi$ and $q_\pi$ can be estimated from experience.

**Bellman Equation**

The Bellman equations express the value of a state in terms of the value of its successor states. They are a consistency condition on the value of states.

$$\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi[G_t|S_t = s] && (16)\\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s] && (17)\\
&= \sum_{a\in\mathcal{A}(s)} \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']\right] && (18)\\
&= \sum_{a\in\mathcal{A}(s)} \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')] && (19)
\end{aligned}$$

The value function $v_\pi$ is the unique solution to its Bellman equation.

## 3.6 Optimal Policies & Optimal Value Functions

We say that $\pi \geq \pi'$ iff $v_\pi(s) \geq v_{\pi'}(s) \quad \forall s \in \mathcal{S}$. The policies that are optimal in this sense are called optimal policies. There may be multiple optimal policies. We denote all of them by $\pi_*$.

The optimal policies share the same optimal value function $v_*(s)$

$$v_*(s) \doteq \max_\pi v_\pi(s) \quad \forall s \in \mathcal{S}. \tag{20}$$

They also share the same optimal action-value function $q_*(s,a)$

$$q_*(s,a) = \max_\pi q_\pi(s,a) \quad \forall s \in \mathcal{S}, a \in \mathcal{A}(s), \tag{21}$$

this is the expected return from taking action $a$ in state $s$ and thereafter following the optimal policy.

$$q_*(s,a) = \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1})|S_t = s, A_t = a]. \tag{22}$$

Since $v_*$ is a value function, it must satisfy a Bellman equation (since it is simply a consistency condition). However, $v_*$ corresponds to a policy that always selects the maximal action. Hence

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')]. \tag{23}$$

Similarly,

$$\begin{aligned}
q_*(s,a) &= \mathbb{E}[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a')|S_t = s, A_t = a] && (24)\\
&= \sum_{s',r} p(s',r|s,a)[r + \gamma \max_{a'} q_*(s',a')]. && (25)
\end{aligned}$$

Note that once one identifies an optimal value function $v_*$, then it is simple to find an optimal policy. All that is needed is for the policy to act greedily with respect to $v_*$. Since $v_*$ encodes all information on future rewards, we can act greedily and still make the long term optimal decision (according to our definition of returns).

Having $q_*$ is even better since we don't need to check $v_*(s')$ in the succeeding states $s'$, we just find $a_* = \mathrm{argmax}_a \, q_*(s, a)$ when in state $s$.