Chapter 4 - Dynamic Programming

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

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

$$q_*(s,a) = E[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s, A_t = a]$$

[4.1] Policy Evaluation:

Computing the state value function v_π for any arbitrary function π is called policy evaluation.

$$v_\pi(s) \doteq E_\pi[G_t \mid S_t = s]$$

$$= E_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s]$$

$$= E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s]$$

$$= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

Consider a sequence of approximate value functions $v_0, v_1, ----$ each mapping S^+ to R. The v_0 is chosen arbitrarily and each successive approximation is obtained by using the Bellman equation for v_π as an update rule:

$$v_{k+1}(s) \doteq E_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s]$$

$$= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_k(s')] \quad \forall s \in S$$

Clearly $v_k = v_\pi$ is a fixed point for this update rule because the Bellman equation for v_π assures us equality.

The sequence $\{v_k\}$ can be shown in general to converge to v_π as $k \to \infty$.

Iterative Policy Evaluation:

Input π, the policy to be evaluated

Algo. parameter $\rightarrow$ θ [to determine accuracy]

Initialise $V(s)\ \forall s \in S^+$, arbitrarily except that $V(\text{terminal}) = 0$

Loop:

$\Delta \leftarrow 0$

Loop for each $s \in S$:

$v \leftarrow V(s)$

$V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma V(s')]$

$\Delta \leftarrow \max(\Delta, |v - V(s)|)$

until $\Delta < \theta$

[4.2] Policy Improvement:

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

$$= \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

Let π & π' be any pair of deterministic policies s.t $\forall s \in S$

$$q_\pi(s, \pi'(s)) \geq v_\pi(s)$$

Then π' must be as good as, or better than, π.

That is
$$\boxed{v_{\pi'}(s) \geq v_\pi(s) \quad \forall s \in S}$$

Let's $\pi'(s) = \pi(s)\ \forall s \in S$ except s' where

$$q_\pi(s', \pi'(s')) > v_\pi(s')$$

then π' is indeed better than π.

Proof:

$$v_\pi(s) \leq q_\pi(s, \pi'(s))$$
$$= E[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = \pi'(s)]$$
$$= E_{\pi'}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s]$$
$$\leq E_{\pi'}[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) \mid S_t = s]$$

$$= E_{\pi'}[R_{t+1} + \gamma E_{\pi'}[R_{t+2} + \gamma q_\pi(S_{t+2}) \mid S_{t+1}, A_{t+1} = \pi'(S_{t+1})] \mid S_t = s]$$
$$= E_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(S_{t+2}) \mid S_t = s]$$
$$\leq E_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 v_\pi(S_{t+3}) \mid S_t = s]$$

$$\leq E_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} \cdots \mid S_t = s]$$

$$= v_{\pi'}(s)$$

$$\pi'(s) \doteq \underset{a}{\operatorname{argmax}}\, q_\pi(s, a)$$
$$= \underset{a}{\operatorname{argmax}}\, E[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a]$$
$$= \underset{a}{\operatorname{argmax}} \sum_{s', r} p(s', r \mid s, a)[r + \gamma v_\pi(s')]$$

The greedy policy takes the action that looks best in the short-term, so we know that it is as good as or better than, the original policy. The process of making a new policy that improves the original policy, by making it greedy w.r.t the value function is called policy improvement.

Suppose, π' is as good as, but not better than π, Then $V_\pi = V_{\pi'}$ and

$$V_{\pi'}(s) = \max_a E[R_{t+1} + \gamma v_{\pi'}(S_{t+1}) \mid S_t = s, A_t = a]$$

$$= \max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_{\pi'}(s)]$$

But this is same as Bellman optimality hence

$V_{\pi'}$ must be V_*

[4.3] <u>Policy Iteration:</u>

Once a policy, π, has been improved using V_π to yield a better policy π', we can then compute $V_{\pi'}$ and improve it again to yield an even better π''.

$$\pi_0 \xrightarrow{E} V_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} V_{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \cdots \xrightarrow{I} \pi_* \xrightarrow{E} V_*$$

<u>Code</u>:

1. Initialisation:
   $V(S) \in R$ and $\pi(S) \in A(S)$ arbitrarily $\forall s \in S$.
2. Policy Evaluation:

```
Loop:
    Δ ← 0
    Loop for each s ∈ S:
        v ← V(s)
        V(s) ← Σ_{s',r} p(s',r|s,π(s)) [r + γ·V(s')]
        Δ ← max(Δ, |v − V(s)|)
until Δ < θ
```

3. Policy Improvement:

policy-stable ← true

For each $s \in S$:

$\quad$ old-action ← $\pi(s)$

$\quad$ $\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s', r | s, a) [r + \gamma V(s')]$

$\quad$ If old-action $\neq \pi(s)$, then policy-stable ← false

If policy-stable, then stop and $V \approx v_*$ and $\pi \approx \pi_*$;

else go to 2.

[4·4] <u>Value Iteration</u>:

Drawback of policy iteration is that each of its iterations involves policy evaluation, which may itself be a protracted iterative computation requiring multiple sweeps through the state set. If policy iteration is done iteratively, its convergence exactly to v_π occurs only in limit.

One important special case of policy evaluation is stopped after just one sweep → its called value iteration

$$v_{k+1}(s) \doteq \max_a E[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s, A_t = a]$$

$$= \max_a \sum_{s',r} p(s', r | s, a) [r + \gamma v_k(s')]$$

Pseudocode:

Loop:

$\Delta \leftarrow 0$

Loop for each $s \in S$:

$v \leftarrow V(s)$

$V(s) \leftarrow \max_a \sum_{s',r} p(s',r|s,a)\left[r+\gamma V(s')\right]$

$\Delta \leftarrow \max(\Delta, |v - V(s)|)$

until $\Delta < \theta$

Outputs a deterministic policy, $\pi \approx \pi_*$, s.t

$$\pi(s) = \operatorname{argmax}_a \sum_{s',r} p(s',r|s,a)\left[r+\gamma V(s')\right]$$