Week 3

Policy

A **policy** π is a mapping from states to probabilities of selecting each possible action:

$$\pi(a \mid s) = \Pr(A_t = a \mid S_t = s)$$

Type	Description
Deterministic	$\pi(s) = a$ — always selects the same action in state s
Stochastic	$\pi(a \mid s)$ — probability distribution over actions

Value Functions

State-Value Function

$$v_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$$

- Expected return starting from state s and following policy π
- Measures how good state s is **under policy** π

Action-Value Function

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$$

- Expected return starting from state s, taking action a, then following policy π
- Measures how good taking action a in state s is **under policy** π

Relationship Between v_π and q_π

From	To	Formula
q_π	v_π	$v_\pi(s) = \sum_a \pi(a \mid s)\, q_\pi(s, a)$
v_π	q_π	$q_\pi(s, a) = \sum_{s', r} p(s', r \mid s, a)\, [r + \gamma v_\pi(s')]$

Value Functions Satisfy Recursive Relationships

Value functions can be expressed recursively — the value of the current state depends on the values of successor states. This recursive structure arises from the definition of return:

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

Since the return G_t is defined as the immediate reward plus the discounted future return, we can write:

$$v_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s]$$

Markov Property Enables Recursive Closure

The key step from G_{t+1} to $v_\pi(S_{t+1})$ relies on the **Markov property**: the distribution of G_{t+1} only depends on the next state S_{t+1}, not on earlier history.

$$\mathbb{E}[G_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s'] = \mathbb{E}[G_{t+1} \mid S_{t+1} = s'] = v_\pi(s')$$

This allows us to substitute G_{t+1} with $v_\pi(S_{t+1})$, yielding the **recursive form**:

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

> The value of a state can be decomposed into the **immediate reward** received after leaving that state, plus the **discounted value** of the successor state.

Component	Meaning
R_{t+1}	Immediate reward after taking action in state s
γG_{t+1}	Discounted future return from successor state s'
$\gamma v_\pi(S_{t+1})$	Discounted value of successor state (by Markov property)

This recursive property is fundamental because:

1. **Enables bootstrapping**: We can estimate the value of a state using estimates of successor states, without waiting for the episode to end
2. **Forms the basis of Bellman equations**: The recursive relationship is formalized into equations that relate values across states
3. **Powers DP and TD methods**: Dynamic programming and temporal-difference learning exploit this structure for efficient computation

Bellman Equations

Bellman Equation for v_π

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

> The value of a state equals the expected immediate reward plus the discounted value of the next state.

Bellman Equation for q_π

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

Backup Diagrams

Diagram	Description
State-value backup	Shows how $v_\pi(s)$ depends on $v_\pi(s')$ for successor states
Action-value backup	Shows how $q_\pi(s, a)$ depends on $q_\pi(s', a')$ for successor state-action pairs

- White circle: state node
- Black dot: action node
- Arcs from state nodes represent policy $\pi(a \mid s)$
- Arcs from action nodes represent dynamics $p(s', r \mid s, a)$

Optimal Value Functions

Optimal State-Value Function

$$v_*(s) = \max_\pi v_\pi(s), \quad \forall s \in \mathcal{S}$$

Optimal Action-Value Function

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

Relationship

$$v_*(s) = \max_a q_*(s, a)$$

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

Bellman Optimality Equations

For v_*

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

For q_*

$$q_*(s, a) = \sum_{s', r} p(s', r \mid s, a) \left[r + \gamma \max_{a'} q_*(s', a')\right]$$

Optimal Policy

A policy π is optimal if $v_\pi(s) \geq v_{\pi'}(s)$ for all states s and all policies π'.

Property	Description
Existence	At least one optimal policy always exists
Shared value	All optimal policies share the same v_* and q_*
Greedy extraction	Given q_*, optimal policy is $\pi_*(s) = \arg\max_a q_*(s, a)$