

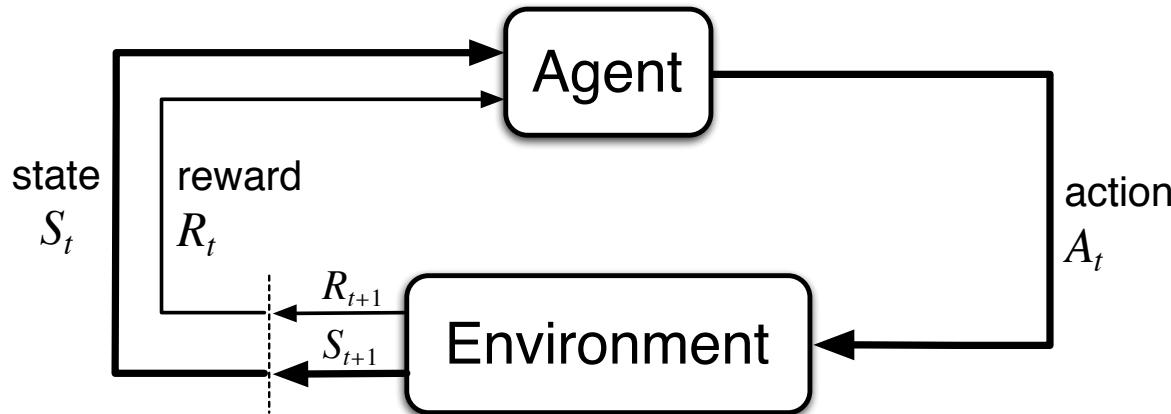
Chapter 3: The Reinforcement Learning Problem

(Markov Decision Processes, or MDPs)

Objectives of this chapter:

- present Markov decision processes—an idealized form of the AI problem for which we have precise theoretical results
- introduce key components of the mathematics: value functions and Bellman equations

The Agent-Environment Interface



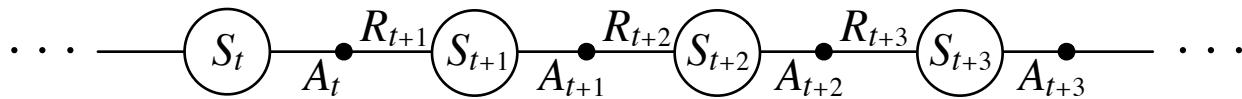
Agent and environment interact at discrete time steps: $t = 0, 1, 2, 3, \dots$

Agent observes state at step t : $S_t \in \mathcal{S}$

produces action at step t : $A_t \in \mathcal{A}(S_t)$

gets resulting reward: $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$

and resulting next state: $S_{t+1} \in \mathcal{S}^+$



Markov Decision Processes

- If a reinforcement learning task has the Markov Property, it is basically a **Markov Decision Process (MDP)**.
- If state and action sets are finite, it is a **finite MDP**.
- To define a finite MDP, you need to give:
 - **state and action sets**
 - one-step “dynamics”

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

$$p(s' | s, a) \doteq \Pr\{S_{t+1} = s' \mid S_t = s, A_t = a\} = \sum_{r \in \mathcal{R}} p(s', r | s, a)$$

$$r(s, a) \doteq \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r | s, a)$$

The Agent Learns a Policy

Policy at step t = π_t =

a mapping from states to action probabilities

$\pi_t(a \mid s)$ = probability that $A_t = a$ when $S_t = s$

Special case - *deterministic policies*:

$\pi_t(s)$ = the action taken with prob=1 when $S_t = s$

- Reinforcement learning methods specify how the agent changes its policy as a result of experience.
- Roughly, the agent's goal is to get as much reward as it can over the long run.

The Markov Property

- By “the state” at step t , the book means whatever information is available to the agent at step t about its environment.
- The state can include immediate “sensations,” highly processed sensations, and structures built up over time from sequences of sensations.
- Ideally, a state should summarize past sensations so as to retain all “essential” information, i.e., it should have the **Markov Property**:

$$\Pr\{R_{t+1} = r, S_{t+1} = s' \mid S_0, A_0, R_1, \dots, S_{t-1}, A_{t-1}, R_t, S_t, A_t\} = \\ p(s', r | s, a) = \Pr\{R_{t+1} = r, S_{t+1} = s' \mid S_t, A_t\}$$

- for all $s' \in \mathcal{S}^+, r \in \mathcal{R}$, and all histories $S_0, A_0, R_1, \dots, S_{t-1}, A_{t-1}, R_t, S_t, A_t$.

The Meaning of Life (goals, rewards, and returns)

Rewards and returns

- The objective in RL is to maximize long-term future reward
- That is, to choose A_t so as to maximize $R_{t+1}, R_{t+2}, R_{t+3}, \dots$
- But what exactly should be maximized?
- The discounted return at time t :

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \dots \quad \text{the discount rate}$$

γ	Reward sequence	Return
0.5(or any)	1 0 0 0...	
0.5	0 0 2 0 0 0...	
0.9	0 0 2 0 0 0...	
0.5	-1 2 6 3 2 0 0 0...	

4 value functions

	state values	action values
prediction	v_π	q_π
control	v_*	q_*

- All theoretical objects, mathematical ideals (expected values)
- Distinct from their estimates:

$$V_t(s) \quad Q_t(s, a)$$

Values are *expected* returns

- The value of a state, given a policy:

$$v_\pi(s) = \mathbb{E}\{G_t \mid S_t = s, A_{t:\infty} \sim \pi\} \quad v_\pi : \mathcal{S} \rightarrow \mathbb{R}$$

- The value of a state-action pair, given a policy:

$$q_\pi(s, a) = \mathbb{E}\{G_t \mid S_t = s, A_t = a, A_{t+1:\infty} \sim \pi\} \quad q_\pi : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$$

- The optimal value of a state:

$$v_*(s) = \max_\pi v_\pi(s) \quad v_* : \mathcal{S} \rightarrow \mathbb{R}$$

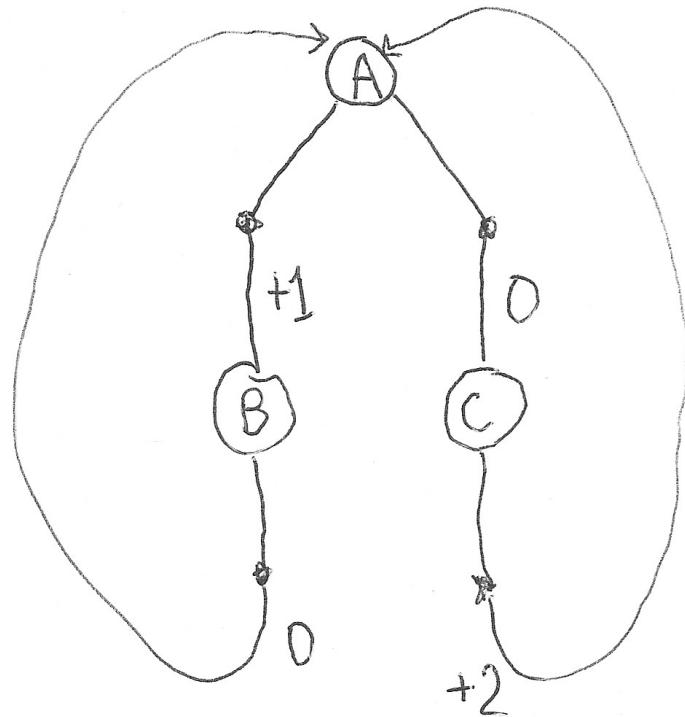
- The optimal value of a state-action pair:

$$q_*(s, a) = \max_\pi q_\pi(s, a) \quad q_* : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$$

- Optimal policy: π_* is an optimal policy if and only if

$$\pi_*(a|s) > 0 \text{ only where } q_*(s, a) = \max_b q_*(s, b) \quad \forall s \in \mathcal{S}$$

- in other words, π_* is optimal iff it is *greedy* wrt q_*



What policy is optimal?

A: Left

B: Right C: Other

If $\gamma=0$?

If $\gamma=.99$

If $\gamma=\frac{1}{2}$?

What we learned last time

- Finite Markov decision processes!
 - States, actions, and rewards
 - And returns
 - And time, discrete time
 - They capture essential elements of life — state, causality
- The goal is to optimize expected returns
 - returns are *discounted sums of future* rewards
- Thus we are interested in *values* — expected returns
- There are four value *functions*
 - state vs state-action values
 - values for a policy vs values for the optimal policy

4 value functions

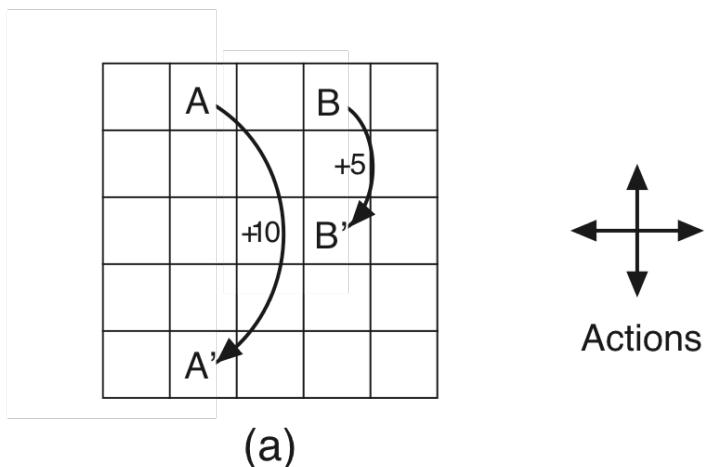
	state values	action values
prediction	v_π	q_π
control	v_*	q_*

- All theoretical objects, mathematical ideals (expected values)
- Distinct from their estimates:

$$V_t(s) \quad Q_t(s, a)$$

Gridworld

- Actions: north, south, east, west; deterministic.
- If would take agent off the grid: no move but reward = -1
- Other actions produce reward = 0, except actions that move agent out of special states A and B as shown.



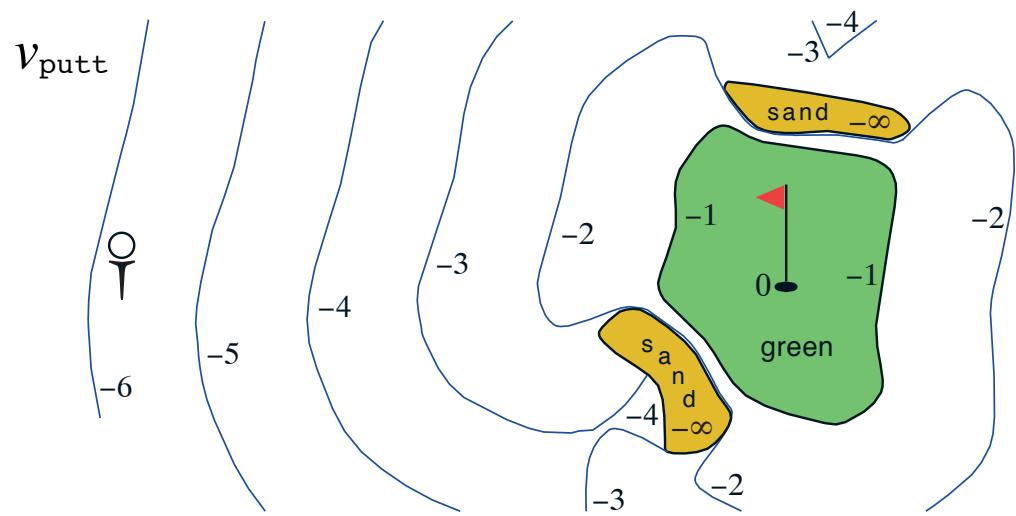
3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

(b)

State-value function
for equiprobable
random policy;
 $\gamma = 0.9$

Golf

- ❑ State is ball location
- ❑ Reward of -1 for each stroke until the ball is in the hole
- ❑ Value of a state?
- ❑ Actions:
 - **putt** (use putter)
 - **driver** (use driver)
- ❑ **putt** succeeds anywhere on the green



Optimal Value Functions

- For finite MDPs, policies can be **partially ordered**:
$$\pi \geq \pi' \quad \text{if and only if } v_\pi(s) \geq v_{\pi'}(s) \text{ for all } s \in \mathcal{S}$$
- There are always one or more policies that are better than or equal to all the others. These are the **optimal policies**. We denote them all π_* .
- Optimal policies share the same **optimal state-value function**:

$$v_*(s) = \max_{\pi} v_{\pi}(s) \quad \text{for all } s \in \mathcal{S}$$

- Optimal policies also share the same **optimal action-value function**:

$$q_*(s, a) = \max_{\pi} q_{\pi}(s, a) \quad \text{for all } s \in \mathcal{S} \text{ and } a \in \mathcal{A}$$

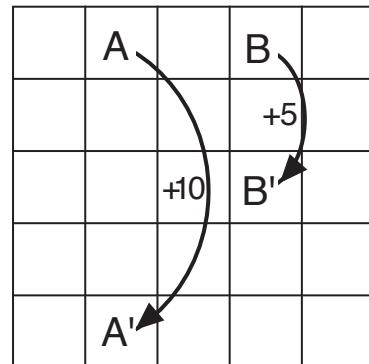
This is the expected return for taking action a in state s and thereafter following an optimal policy.

Why Optimal State-Value Functions are Useful

Any policy that is greedy with respect to v_* is an optimal policy.

Therefore, given v_* , one-step-ahead search produces the long-term optimal actions.

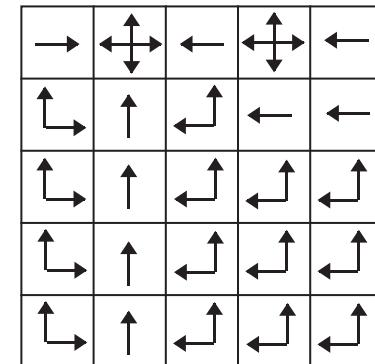
E.g., back to the gridworld:



a) gridworld

22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

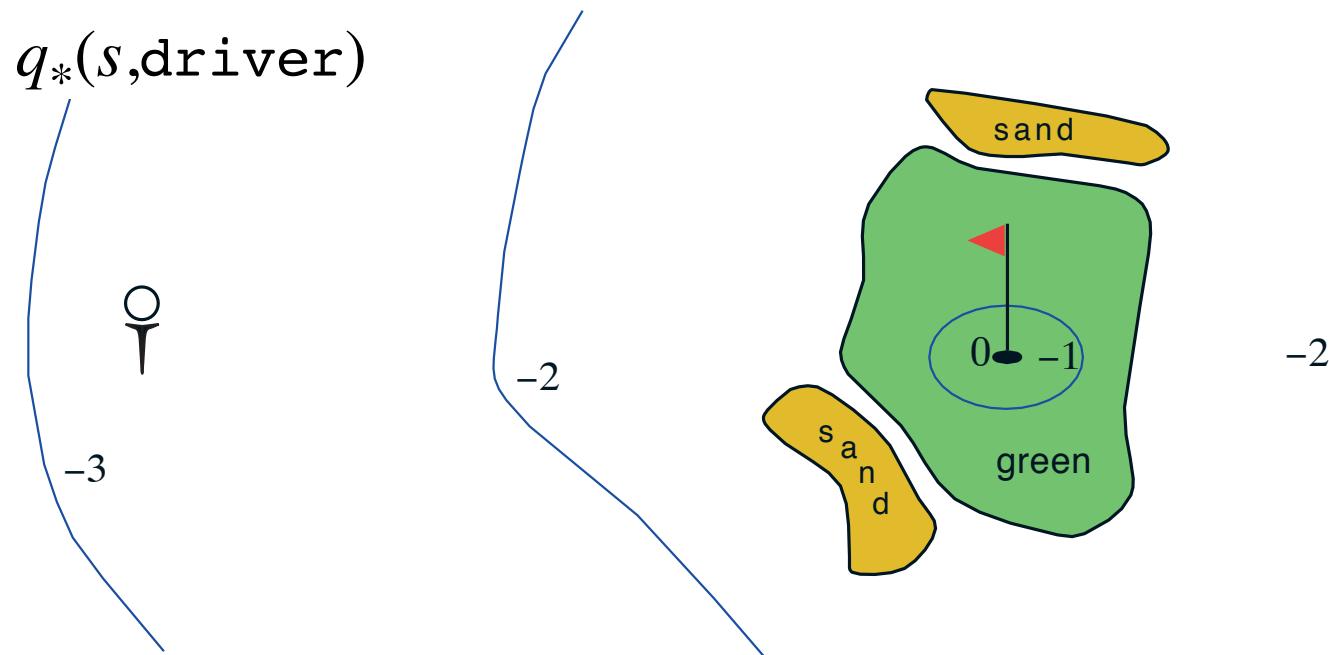
b) v_*



c) π_*

Optimal Value Function for Golf

- We can hit the ball farther with `driver` than with `putter`, but with less accuracy
- $q_*(s, \text{driver})$ gives the value of using `driver` first, then using whichever actions are best



What About Optimal Action-Value Functions?

Given q_* , the agent does not even have to do a one-step-ahead search:

$$\pi_*(s) = \arg \max_a q_*(s, a)$$

Value Functions

x 4

Bellman Equations

x 4

Bellman Equation for a Policy π

The basic idea:

$$\begin{aligned} G_t &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \dots \\ &= R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \dots) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned}$$

So:

$$\begin{aligned} v_\pi(s) &= E_\pi \left\{ G_t \mid S_t = s \right\} \\ &= E_\pi \left\{ R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s \right\} \end{aligned}$$

Or, without the expectation operator:

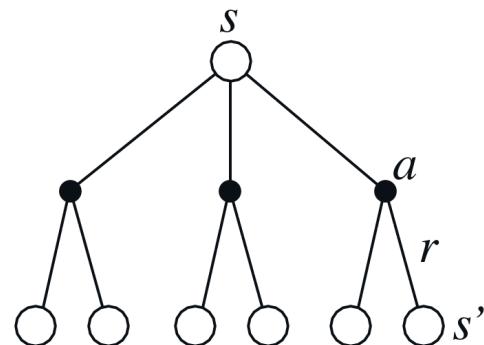
$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r | s, a) [r + \gamma v_\pi(s')]$$

More on the Bellman Equation

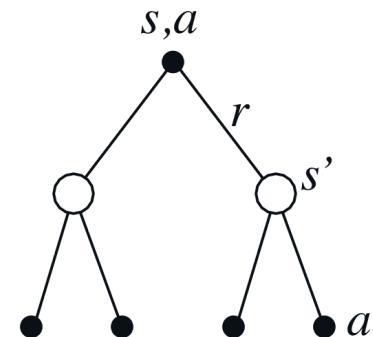
$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) [r + \gamma v_\pi(s')]$$

This is a set of equations (in fact, linear), one for each state. The value function for π is its unique solution.

Backup diagrams:



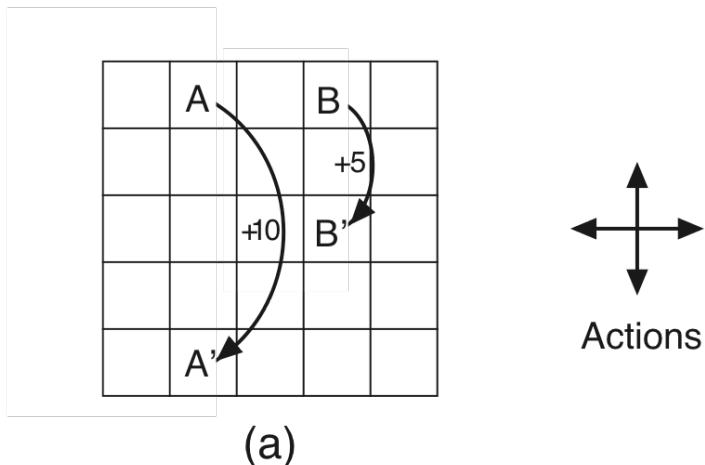
for v_π



for q_π

Gridworld

- Actions: north, south, east, west; deterministic.
- If would take agent off the grid: no move but reward = -1
- Other actions produce reward = 0, except actions that move agent out of special states A and B as shown.



3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

(b)

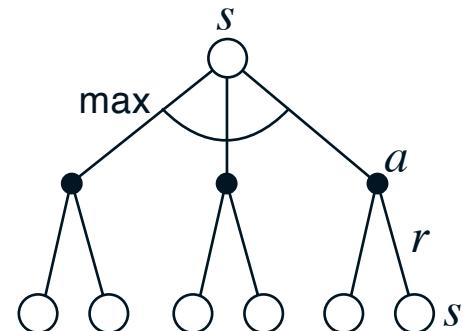
State-value function
for equiprobable
random policy;
 $\gamma = 0.9$

Bellman Optimality Equation for v_*

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

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

The relevant backup diagram:

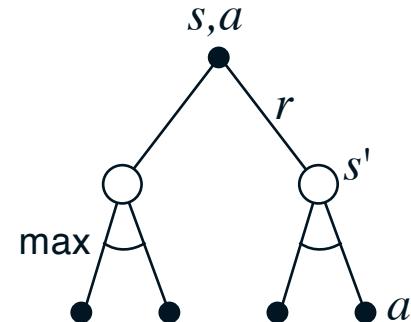


v_* is the unique solution of this system of nonlinear equations.

Bellman Optimality Equation for q_*

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

The relevant backup diagram:



q_* is the unique solution of this system of nonlinear equations.

Solving the Bellman Optimality Equation

- Finding an optimal policy by solving the Bellman Optimality Equation requires the following:
 - accurate knowledge of environment dynamics;
 - we have enough space and time to do the computation;
 - the Markov Property.
- How much space and time do we need?
 - polynomial in number of states (via dynamic programming methods; Chapter 4),
 - BUT, number of states is often huge (e.g., backgammon has about 10^{20} states).
- We usually have to settle for approximations.
- Many RL methods can be understood as approximately solving the Bellman Optimality Equation.

Summary

- Agent-environment interaction
 - States
 - Actions
 - Rewards
- Policy: stochastic rule for selecting actions
- Return: the function of future rewards agent tries to maximize
- Episodic and continuing tasks
- Markov Property
- Markov Decision Process
 - Transition probabilities
 - Expected rewards
- Value functions
 - State-value function for a policy
 - Action-value function for a policy
 - Optimal state-value function
 - Optimal action-value function
- Optimal value functions
- Optimal policies
- Bellman Equations
- The need for approximation