Markov Decision Processes

Return, Discounting, and Value Functions

Instructor: Sarwan Ali

Department of Computer Science
Georgia State University

Understanding Returns and Value Functions

Today's Learning Journey

Markov Decision Process (MDP) - Quick Review

MDP Definition

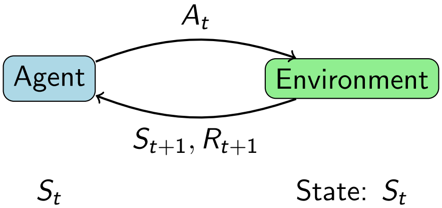
An MDP is a 5-tuple: $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$: Set of states
- $\mathcal{A}$: Set of actions
- $\mathcal{P}$: Transition probability function
- $\mathcal{R}$: Reward function
- γ: Discount factor

Key Properties

- **Markov Property**: Future depends only on current state
- **Sequential Decision Making**: Actions affect future states
- **Stochastic Outcomes**: Uncertainty in transitions and rewards

The Agent-Environment Interaction



Interaction Sequence

At each time step t:

1. Agent observes state S_t
2. Agent selects action A_t
3. Environment returns reward R_{t+1} and next state S_{t+1}

What is Return?

Definition: Return

The **return** G_t is the total accumulated reward from time step t onwards:

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \cdots = \sum_{k=0}^{\infty} R_{t+k+1} \tag{1}$$

Problem: Infinite Horizons

For infinite horizons, this sum may diverge!

Example: Simple Chain

Consider states $S_1 \to S_2 \to S_3$ with rewards $+1, +1, +1, \ldots$

Without discounting: $G_0 = 1 + 1 + 1 + \cdots = \infty$

Types of Tasks

Episodic Tasks

- Have natural ending (terminal states)
- Examples: Games, robot navigation
- Return: $G_t = \sum_{k=0}^{T-t-1} R_{t+k+1}$
- T = terminal time step

Continuing Tasks

- No natural ending
- Examples: Process control, trading
- Need discounting to ensure convergence
- Return: $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

Key Insight

Discounting allows us to handle both episodic and continuing tasks in a unified framework!

The Discount Factor γ

Discounted Return

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \quad (2)$$

- $\gamma \in [0,1]$ is the **discount factor**
- $\gamma = 0$: Only immediate reward matters (myopic)
- $\gamma = 1$: All future rewards equally important
- $\gamma \in (0,1)$: Gradually decreasing importance of future rewards

Mathematical Convergence

If $|R_t| \leq R_{\max}$ for all t, then:

$$|G_t| \leq \sum_{k=0}^{\infty} \gamma^k R_{\max} = \frac{R_{\max}}{1-\gamma}$$

Why Discount?

Mathematical Reasons

- Ensures convergence
- Makes problems well-defined
- Enables recursive relationships

Computational Reasons

- Finite value functions
- Tractable optimization
- Stable algorithms

Practical Reasons

- **Uncertainty**: Future is uncertain
- **Time preference**: Immediate rewards preferred
- **Modeling**: Approximates real-world scenarios

Real-world Example

$100 today vs $100 in 10 years?

Recursive Property of Return

Key Insight

The return satisfies a recursive relationship:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \quad (3)$$

$$= R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \cdots) \quad (4)$$

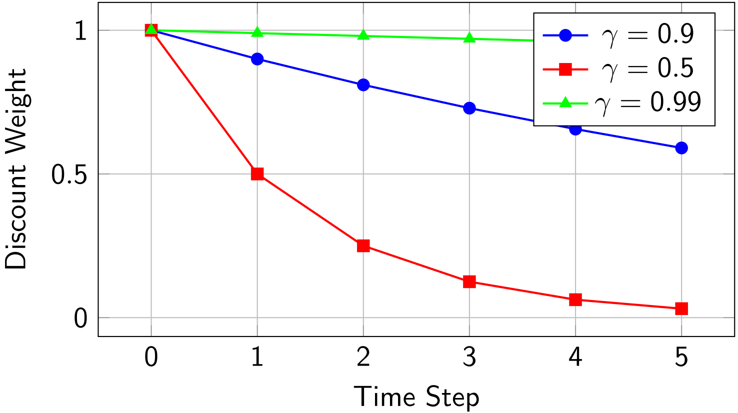
$$= R_{t+1} + \gamma G_{t+1} \quad (5)$$

This is Fundamental!

This recursive property is the foundation for:

- Bellman equations
- Dynamic programming
- Temporal difference learning

Discount Factor Impact Visualization



Observation: Higher γ values give more weight to future rewards.

State Value Function

Definition: State Value Function

The **state value function** $v_\pi(s)$ under policy π is the expected return starting from state s:

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

where $\mathbb{E}_\pi[\cdot]$ denotes expectation under policy π.

Expanded Form

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

Interpretation

$v_\pi(s)$ tells us "how good" it is to be in state s when following policy π.

Action Value Function

Definition: Action Value Function

The **action value function** $q_\pi(s, a)$ under policy π is the expected return starting from state s, taking action a:

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

Expanded Form

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

Interpretation

$q_\pi(s, a)$ tells us "how good" it is to take action a in state s when following policy π.

Relationship Between Value Functions

From Action Values to State Values

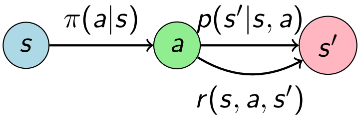
$$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s, a)$$

The value of a state is the expected value over all possible actions, weighted by the policy.

From State Values to Action Values

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

The value of an action is the expected immediate reward plus the discounted value of the next state.



The Bellman Equation for v_π

Bellman Equation

The state value function satisfies the Bellman equation:

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s] && (6) \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s] && (7) \end{aligned}$$

Expanded Form

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

Key Insight

This is a system of linear equations! For n states, we have n equations in n unknowns.

The Bellman Equation for q_π

Bellman Equation for Action Values

The action value function satisfies:

$$q_\pi(s, a) = \mathbb{E}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a] \tag{8}$$

$$= \sum_{s'} p(s'|s, a)[r(s, a, s') + \gamma v_\pi(s')] \tag{9}$$

Alternative Form

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

Matrix Form

These equations can be written compactly using matrices, enabling efficient computation.

Example: Simple Grid World

S1	S2	S3
S4	S5	S6
S7	S8	+10

Setup

- 3×3 grid world
- Goal: Reach bottom-right (+10 reward)
- Actions: Up, Down, Left, Right
- Other transitions: -1 reward
- $\gamma = 0.9$

Policy

Uniform random policy: $\pi(a|s) = 0.25$ for all a

Question

What are the value functions $v_\pi(s)$ for each state?

Grid World Solution Process

Bellman Equation Setup

For each state s, we have:

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

Example for State S5 (center)

$$v_\pi(S5) = 0.25 \times [(-1 + 0.9 v_\pi(S2)) + (-1 + 0.9 v_\pi(S8)) \quad (10)$$
$$+ (-1 + 0.9 v_\pi(S4)) + (-1 + 0.9 v_\pi(S6))] \quad (11)$$

Solution Method

- Set up system of 8 linear equations
- Solve using matrix methods or iteration
- Terminal state: $v_\pi(\text{Goal}) = 0$

Computing Value Functions: Methods

Direct Solution

Solve linear system:

$$\mathbf{v} = \mathbf{r} + \gamma \mathbf{P}\mathbf{v}$$

$$\mathbf{v} = (\mathbf{I} - \gamma \mathbf{P})^{-1}\mathbf{r}$$

Complexity: $O(n^3)$ for n states

Iterative Methods

Value Iteration:

$$v_{k+1}(s) = \sum_a \pi(a|s) \sum_{s'} p(s'|s,a)[r + \gamma v_k(s')]$$

Complexity: $O(n^2)$ per iteration

Practical Considerations

- Direct solution for small state spaces ($n < 1000$)
- Iterative methods for large state spaces
- Convergence guaranteed for $\gamma < 1$

Optimal Value Functions

Optimal State Value Function

$$v^*(s) = \max_\pi v_\pi(s)$$

The maximum value achievable in state s over all possible policies.

Optimal Action Value Function

$$q^*(s, a) = \max_\pi q_\pi(s, a)$$

The maximum value achievable by taking action a in state s and then following the optimal policy.

Fundamental Relationship

$$v^*(s) = \max_a q^*(s, a)$$

Bellman Optimality Equations

For State Values

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

For Action Values

$$q^*(s,a) = \sum_{s'} p(s'|s,a)[r(s,a,s') + \gamma \max_{a'} q^*(s',a')]$$

Key Difference

These are **nonlinear** equations due to the max operator!

Solution Methods

- Value Iteration
- Policy Iteration
- Linear Programming

Key Takeaways

Return and Discounting

- Return G_t measures total future reward
- Discount factor γ balances immediate vs. future rewards
- Enables unified treatment of episodic and continuing tasks

Value Functions

- $v_\pi(s)$: Expected return from state s under policy π
- $q_\pi(s, a)$: Expected return from state-action pair (s, a)
- Satisfy recursive Bellman equations

Optimal Value Functions

- $v^*(s)$ and $q^*(s, a)$: Best possible performance
- Satisfy Bellman optimality equations
- Foundation for finding optimal policies

Mathematical Summary

Core Equations

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{12}$$

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] \tag{13}$$

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \tag{14}$$

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s'} p(s'|s, a)[r + \gamma v_\pi(s')] \tag{15}$$

$$v^*(s) = \max_a \sum_{s'} p(s'|s, a)[r + \gamma v^*(s')] \tag{16}$$

Next Steps

Coming Up

- **Dynamic Programming**: Value and Policy Iteration algorithms
- **Monte Carlo Methods**: Learning from experience
- **Temporal Difference Learning**: Combining DP and MC
- **Function Approximation**: Handling large state spaces

Homework/Practice

- Solve small grid world problems by hand
- Implement value iteration algorithm
- Experiment with different discount factors
- Analyze convergence properties

Questions?

Discussion and Clarifications

Contact: sali85@student.gsu.edu

Course Materials:
https://sarwanpasha.github.io/Courses/Reinforcement_Learning/int_RL.html