Minor in AI

Revision

Reinforcement Learning

May 08, 2025

1 Grid World Example in Reinforcement Learning

1.1 Environment Description

The grid world is a finite, stochastic environment composed of:

- **Normal cells**: neutral transitions.
- **Goal cell**: rewards +1.
- **Poison cell**: rewards -1.
- **Wall cell**: impassable.

1.2 Transition Model

The environment is **stochastic**, meaning the intended action might result in moving in another direction with some probability. For example:

$$P(s'|s,a) = \begin{cases} 0.8 & \text{intended direction} \\ 0.1 & \text{left of intended} \\ 0.1 & \text{right of intended} \end{cases}$$

1.3 Objective: Compute Optimal Value Function $V^*(s)$

We want to find:

$$V^*(s) = \max_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V^*(s')]$$

This is known as the **Bellman Optimality Equation**.

1.4 Value Iteration Algorithm

1. Initialize $V(s) = 0$ for all s.
2. Repeat until convergence:
   - For each state s: $V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V_k(s')]$
3. Stop when $\max \_s|V\_k + 1(s) - V\_k(s)| < \epsilon$.
4. Extract policy: $\pi^*(s) = \arg\max_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V(s')]$

1.5 Visualization

			+1
-1			

2 Policy Iteration

2.1 Two-step Algorithm

1. **Policy Evaluation**: For fixed π, compute V^π: $V^\pi(s) = \sum_{s'} P(s'|s, \pi(s))[R(s, \pi(s), s') + \gamma V^\pi(s')]$
2. **Policy Improvement**: Update policy: $\pi'(s) = \arg\max_a \sum_{s'} P(s'|s, a)[R(s, a, s') + \gamma V^\pi(s')]$
3. Repeat until policy is stable.

3 Recommendation System as RL Problem

3.1 Problem Setup

- State: (N, S) = slots left N, set of available movies S.
- Action: Pick a movie $m \in S$, recommend without repetition.
- Transition: Based on probabilistic user click behavior.
- Reward: Click = 1, No Click = 0.

3.2 Dynamic Programming Formulation

Let $V(N, S)$ be expected cumulative reward (clicks):

$$V(N, S) = \max_{a \in S} \sum_{r \in \{0,1\}} P(r|a)[r + V(N-1, S \setminus \{a\})]$$

Base case: $V(0, \cdot) = 0$.

3.3 Algorithm Overview

1. Generate all subsets of movies.
2. Initialize value for $N = 0$.
3. Iteratively compute $V(N, S)$ using DP.
4. Derive optimal sequence of recommendations.

4 Monte Carlo (MC) Methods in RL

4.1 Characteristics

- Model-free: no need for transition probabilities.
- Based on episodes: from start to terminal state.
- Returns are defined recursively: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$

4.2 Types of Monte Carlo Methods

- **MC Prediction**: Estimate $V^\pi(s)$: $V(s) = \frac{1}{N(s)} \sum_{i=1}^{N(s)} G^{(i)}$
- **MC Control**: Improve policy using **Exploring Starts** or ε**-greedy** policies.

Aspect	**Dynamic Programming (DP)**	**Monte Carlo (MC)**
Model	Requires full model	Model-free
Computation	Bellman updates	Averaging returns
Use Case	Known models, small state space	Episodic tasks, unknown models
Dependence	Bootstraps from $V(s')$	Uses complete episodes

5 Case Study : Course Enrollment

Consider a student enrolled in a two-year postgraduate program. At each semester (state), the student must choose between taking a core course or an elective course (action). Electives may be more enjoyable or relevant to personal interests, while core courses may be more aligned with industry expectations.

- Each course choice results in certain rewards.
- Immediate rewards could be grades, skills learned, or networking opportunities.
- Delayed rewards include internship opportunities, placement offers, or graduate school admits.

The student's **objective** is to maximize their career outcome at graduation — the terminal state. Career outcome is influenced cumulatively by the student's past choices.

5.1 Challenges

The student doesn't know in advance which courses will yield the best long-term reward.

- Some courses may seem unappealing (low immediate reward) but are beneficial long-term (high future reward).
- Career outcome is uncertain and based on multiple factors.

This situation perfectly illustrates sequential decision-making under uncertainty, which is the heart of Reinforcement Learning.

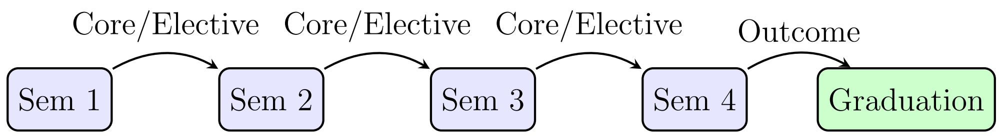
5.2 Formalization as a Markov Decision Process (MDP)

We can formalize the course enrollment scenario as an MDP:

- **States** ($\mathcal{S}$): $s_t \in$ Year 1 Sem 1, Year 1 Sem 2, $\ldots$, Graduation — represents the academic stage.
- **Actions** ($\mathcal{A}$): $a_t \in$ Core, Elective — course type selected at time t.
- **Transition Function** ($\mathcal{P}(s'|s, a)$): Probability of moving to the next academic state s' given current state s and action a (usually deterministic in course structure).

- **Reward Function** ($\mathcal{R}(s, a)$): Real-valued function capturing immediate reward — e.g., +5 for high grades, −2 for stress, +10 for relevant internship.
- **Policy** ($\pi(a|s)$): Strategy to choose courses — could be random initially, later optimized to maximize expected cumulative rewards.
- **Return** (G_t): $G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots + \gamma^{T-t-1} r_T$ — cumulative discounted reward up to graduation.
- **Objective**: Find the optimal policy π^* that maximizes $\mathbb{E}_\pi[G_t]$ for all s_t.

5.3 Environment Diagram



6 Key Takeaways

1. **RL Framework:** Reinforcement Learning involves agents taking actions in an environment to maximize cumulative reward, defined by states, actions, rewards, and policies.
2. **Dynamic Programming (DP):** Uses a known transition model to compute optimal value functions and policies via value iteration and policy iteration.
3. **Stochasticity in Grid World:** Actions may lead to multiple outcomes with probabilities, requiring expected value calculations in updates.
4. **Monte Carlo (MC) Methods:** Model-free approach that learns value functions using returns from complete episodes, suitable for unknown or complex environments.
5. **DP vs MC:** DP uses bootstrapping and requires a model; MC relies on sample episodes and does not require model knowledge.