Reinforcement Learning Lecture Notes

Guo Yuanxin

November 2019

This note is based on Emma Brunskill's lecturing of **CS234: Reinforcement Learning** at Stanford University.

1 Lecture 1: Introduction

1.1 Overview of Course

What is reinforcement learning?

Use one sentence to summarize:

Learn to make good sequences of decisions.

This terse sentence encodes three key points: 1) **Sequence of decisions**; 2) **Good**; and 3) **Learn**. The fundamental challenge of AI and machine learning is to make good decisions under uncertainty.

What does reinforcement learning involve?

Optimization. Finding an optimal or near-optimal way of making decisions is our goal.

Delayed consequences. Decisions now can impact things much later. A challenge arise: How to figure out the causal relationship between decisions made and outcomes in the future?

Exploration. The only way to learn is by making decisions. The challenge here is the "censored data". We only get a reward for what we have tried. In other words, our decisions impact what we learn about.

Generalization. A natural question is why we don't pre-program a policy. RL provides a way by some high level representation where we can generalize our solution even if we encounter a problem we have never seen before.

1.2 Introduction to Sequential Decision Making under Uncertainty

Modeling the Decision Process We model the sequential decision making as an interactive, close-loop process. Agent take actions, affecting the world, which give back observations and a reward. Our goal is to select actions to maximize total **expected** future reward. We emphasize **expected** as the world is stochastic. Note that this is not always be the right criterion.

Two challenges are of our concern: 1) How to balance immediate and long term rewards? and 2) How to choose a strategy to achieve high rewards?

We now formulate the discrete time sequential decision process. At each time step t:

- Agent takes an action a_t
- World updates given a_t, emits observation o_t and reward r_t
- Agent receives observation o_t and reward r_t

The history h_t is a tuple of all information in the past:

$$h_t = (a_1, o_1, r_1, \ldots, a_t, o_t, r_t).$$

The agent chooses action based on history.

Another important thing is the state. It is the information assumed to determine what happens next.

What should be noted is that there are two notions of state. One is a notion of "world state" which determines how the world works. But to an RL agent, this is always unknown. Also, it is often that it includes unnecessary information to the agent.

Another is the agent state, which is what the agent uses to make decisions about how to act. It is generally a function of the history: $s_t = f(h_t)$.

Markovity We are going to use **Markov assumption** a lot in this class. A state is Markov iff:

$$p(s_{t+1}|s_t, a_t) = p(s_{t+1}|h_t, a_t).$$

This implies the current state is a **sufficient statistic** of history. Another classical way to interpreting this is to say: Future is independent of past given present.

In a lot of applications, the scenarios seem to be non-Markov. But the popularity of the Markov assumption is that it can be *made* Markov be setting the state as history. In practice, we usually assume the most recent observation is the sufficient statistic of history: $s_t = o_t$. Reasonably representing the state has big implications for computational complexity, data required, and resulting performance.

Types of Sequential DP By how we represent the state, we classify the decision processes:

- Full Observability / Markov Decision Process (MDP)
  Environment and world state are the same: $s_t = o_t$.

- Partial Observability / Partial Observable Markov Decision Process (POMDP)
  Agent state is not the same as the world state. Agent constructs its own state by using history $s_t = h_t$, or his beliefs, or NN, ...
  (e.g. Poker: only observe one's own cards)

We now compare the decision processes in a larger framework:

- Bandits: Actions have no influence on next observations. No delayed rewards.

- MDPs & POMDPs: Actions influence future observations. Might need credit assignment and strategic actions.

Another important question is how the world changes. It is usually modeled using either a deterministic model or a stochastic model.

- Deterministic: Given history & action, single observation & reward. Commonly used in robotics & control.

- Stochastic: Given history & action, many potential observations & rewards. Commonly assumed for patients, customers, etc.

RL Algorithm An essential part of an RL algorithm is the model. A model is basically the agent's representation of how the world changes in response to agent's action. For a MDP, the **transition model** gives the distribution of the next state given current state and action:

$$p(s_{t+1} = s' | s_t = s, a_t = a).$$

There is also a reward model predicting the *expected reward* of taking an action at a certain state:

$$r(s_t = s, a_t = a) = \mathbb{E}[r_t | s_t = s, a_t = a].$$