MDPs: model-based methods

Model-Based Value Iteration

Data: $s_0; a_1, r_1, s_1; a_2, r_2, s_2; a_3, r_3, s_3; \ldots; a_n, r_n, s_n$

Key idea: model-based learning

Estimate the MDP: $T(s, a, s')$ and $\text{Reward}(s, a, s')$

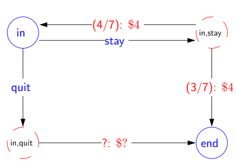
Transitions:

$$\hat{T}(s, a, s') = \frac{\#\text{ times } (s, a, s') \text{ occurs}}{\#\text{ times } (s, a) \text{ occurs}}$$

Rewards:

$$\widehat{\text{Reward}}(s, a, s') = r \text{ in } (s, a, r, s')$$

Model-Based Value Iteration



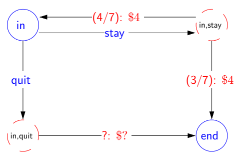
Data (following policy $\pi(s) = \text{stay}$):

[in; stay, 4, end]

- Estimates converge to true values (under certain conditions)
- With estimated MDP $(\hat{T}, \widehat{\text{Reward}})$, compute policy using value iteration

- The first idea is called **model-based** value iteration, where we try to estimate the model (transitions and rewards) using Monte Carlo simulation.
- Monte Carlo is a standard way to estimate the expectation of a random variable by taking an average over samples of that random variable.
- Here, the data used to estimate the model is the sequence of states, actions, and rewards in the episode. Note that the samples being averaged are not independent (because they come from the same episode), but they do come from a Markov chain, so it can be shown that these estimates converge to the expectations by the ergodic theorem (a generalization of the law of large numbers for Markov chains).
- But there is one important caveat...

Problem



Problem: won't even see (s, a) if $a \neq \pi(s)$ ($a = $ quit)

Key idea: exploration

To do reinforcement learning, need to explore the state space.

Solution: need π to **explore** explicitly (more on this later)

- So far, our policies have been deterministic, mapping s always to $\pi(s)$. However, if we use such a policy to generate our data, there are certain (s, a) pairs that we will never see and therefore never be able to estimate their Q-value and never know what the effect of those actions are.
- This problem points at the most important characteristic of reinforcement learning, which is the need for **exploration**. This distinguishes reinforcement learning from supervised learning, because now we actually have to act to get data, rather than just having data poured over us.
- To close off this point, we remark that if π is a non-deterministic policy which allows us to explore each state and action infinitely often (possibly over multiple episodes), then the estimates of the transitions and rewards will converge.
- Once we get an estimate for the transitions and rewards, we can simply plug them into our MDP and solve it using standard value or policy iteration to produce a policy.
- Notation: we put hats on quantities that are estimated from data ($\hat{Q}^*, \hat{T}$) to distinguish from the true quantities (Q^*, T).

From model-based to model-free

$$\hat{Q}_{\text{opt}}(s, a) = \sum_{s'} \hat{T}(s, a, s')[\widehat{\text{Reward}}(s, a, s') + \gamma \hat{V}_{\text{opt}}(s')]$$

All that matters for prediction is (estimate of) $Q_{\text{opt}}(s, a)$.

Key idea: model-free learning

Try to estimate $Q_{\text{opt}}(s, a)$ directly.

- Taking a step back, if our goal is to just find good policies, all we need is to get a good estimate of $\hat{Q}_{\text{opt}}$. From that perspective, estimating the model (transitions and rewards) was just a means towards an end. Why not just cut to the chase and estimate $\hat{Q}_{\text{opt}}$ directly? This is called **model-free** learning, where we don't explicitly estimate the transitions and rewards.