Reinforcement Learning

Model-Free Reinforcement Learning

aka "Playing without the rules"

Akka Zemmari

About today

Learning without knowing the rules

MDP is not fully known.

- Set of states S and actions A are *known*
- $p(s', r|s, a)$ is *unknown*

Wanted

- **model-free prediction**. Evaluate the value function in an unknown MDP
- **model-free control**. Optimize the value function in an unknown MDP

Plan of the day

1. **Monte Carlo methods**
2. **Temporal Difference methods**
   - 2.1 **SARSA** (on-policy learning)
   - 2.2 **Q-learning** (off-policy learning)
   - 2.3 **Double Q-learning**

Monte Carlo methods

Intuition about Monte Carlo methods

See blackboard for discussion on the topic (law of large numbers).

Recap last lectures

Optimal Bellman Equations

- State-Value function:

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

- Action-Value function:

$$\forall s \in S, a \in A, Q^*(s, a) = \sum_{s',r} p(s', r|s, a)[r + \gamma V^*(s')]$$

Optimal policy

- $\pi^*(s) = \arg\max_a Q^*(s, a)$
- with the form of the policy improvement step:

$$\pi^{k+1}(s) = \arg\max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V^{\pi^k}(s')]$$

Recap last lectures

Optimal Bellman Equations

- State-Value function:

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

- Action-Value function:

$$\forall s \in S, a \in A, Q^*(s, a) = \sum_{s',r} p(s', r|s, a)[r + \gamma V^*(s')]$$

Optimal policy

- $\pi^*(s) = \arg\max_a Q^*(s, a)$
- with the form of the policy improvement step:

$$\pi^{k+1}(s) = \arg\max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V^{\pi^k}(s')]$$

Problem: All these equations are model-based, since they require the knowledge of the transition probabilities $p(s', r|s, a)$.

Recap last lectures

How can we turn these model-based equations into model-free ones?
The expressions can be rewritten as:

- State-Value function:

$$\forall s \in S, V^*(s) = \max_a \mathbb{E}[G_t | S_t = s, A_t = a]$$

- Action-Value function:

$$\forall s \in S, a \in A, Q^*(s, a) = \mathbb{E}[G_t | S_t = s, A_t = a]$$

Monte Carlo policy evaluation

We want to evaluate the value function of a policy π.

Most natural idea. Monte Carlo.

Sample N trajectories from s_0 using policy π:

i-th trajectory: $s_0^i = s_0, a_0^i, r_0^i, s_1^i, a_1^i, r_1^i, s_2^i, a_2^i, r_2^i, \ldots$

i-th total reward. $G^i = r_0^i + \gamma r_1^i + \gamma^2 r_2^i + \ldots$

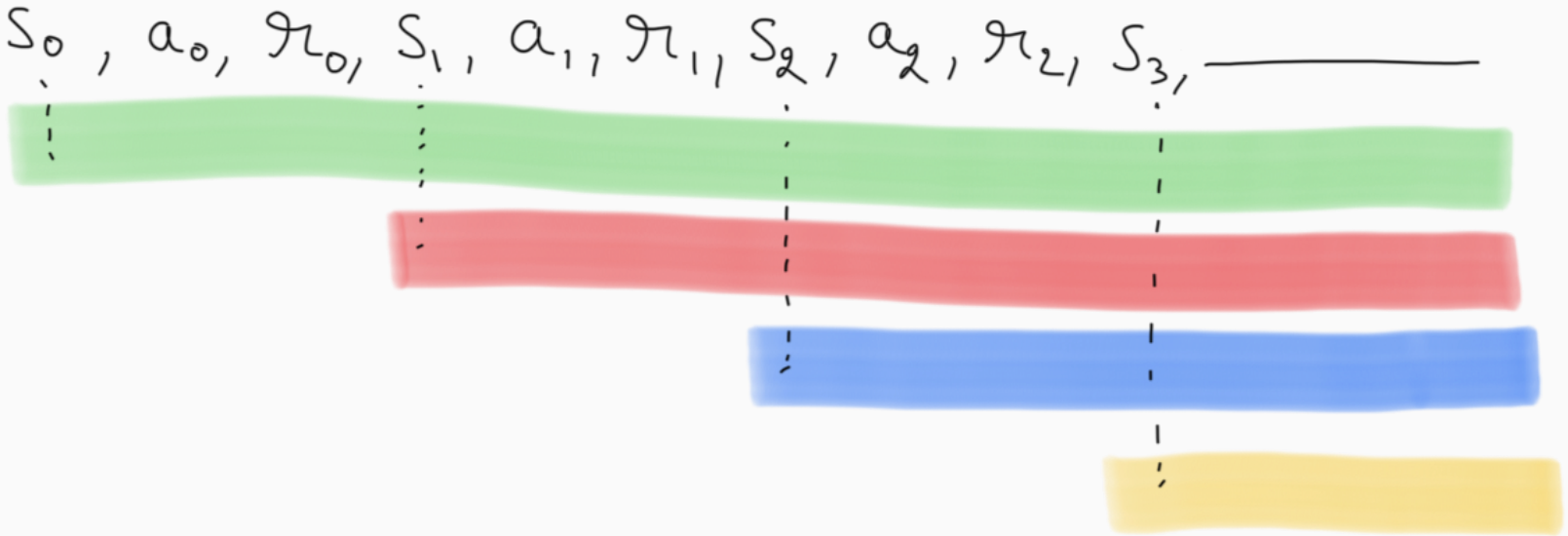
Then, we can compute the empirical average of the returns:

$$\widehat{V}^\pi(s_0) = \frac{1}{N} \sum_{i=1}^{N} G^i$$

Monte Carlo policy evaluation

What about others states? $s \neq s_0$?

Idea. One trajectory $\Rightarrow$ many trajectories from different starting states.



Monte Carlo policy evaluation

Question

$\widehat{V}^{\pi}(s) =$ average of total reward over all trajectories

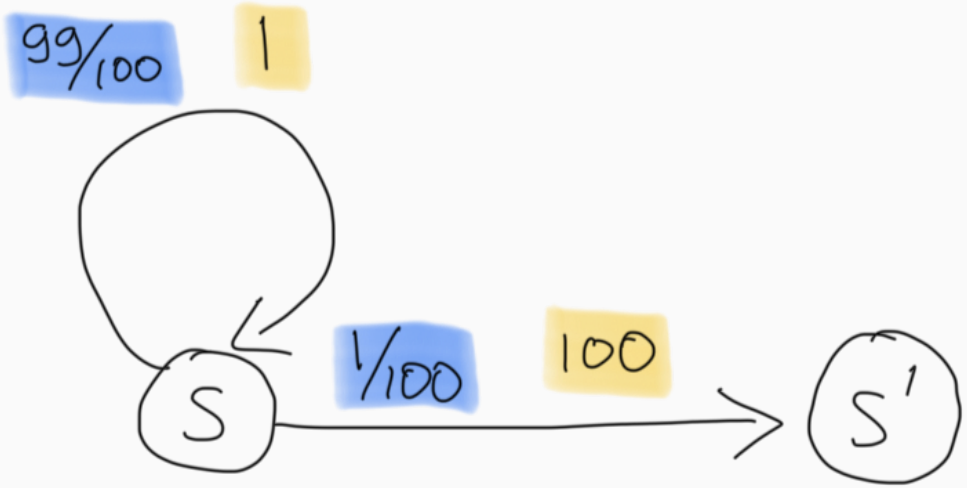
1. which start in the first occurrence of s?
2. which start in any occurrence of s?
3. which start in the last occurrence of s?

Monte Carlo policy evaluation

Answer

- 1. and 2. converges to $V^{\pi}(s)$ for all state s visited by the policy π
- 3. does not.

Counter-example to 3.



Monte Carlo policy evaluation

Which one to use

$\widehat{V}^{\pi}(s) =$ average of total reward over all trajectories

1. which start in the first occurrence of s?
2. which start in any occurrence of s?
3. which start in the last occurrence of s?

We will use 2. because it is easier to compute and give more data.

Algorithm - Monte Carlo policy evaluation

1. **Input**: policy π
2. **Initialize**:
   - $V(s) = 0$ for all $s \in S$
   - $N(s) = 0$ for all $s \in S$ (number of visits to s)
3. **Repeat**:
   - Generate a trajectory $s_0, a_0, r_0, s_1, a_1, r_1, \ldots s_T, a_T, r_T$ using π
   - For all time steps $t \leq T$:
     - $G = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots$
     - $N(s_t) = N(s_t) + 1$
     - $V(s_t) = V(s_t) + \frac{1}{N(s_t)}(G - V(s_t))$

Complexity question.

It takes $O(T^2)$ time for one trajectory.

Can we do better?

Algorithm - Monte Carlo policy evaluation

Trick. Going backward in time.

- $G = 0$
- For all time steps $t = T, T-1, \ldots, 0$:
  - $G = r_t + \gamma G$
  - $N(s_t) = N(s_t) + 1$
  - $V(s_t) = V(s_t) + \frac{1}{N(s_t)}(G - V(s_t))$

This time, it takes $O(T)$ time for one trajectory.

Still one last problem

It converges only for states that are visited by the policy π. We might miss a lot of them.

Showtime

Monte Carlo Policy Evaluation with ε-Greedy Exploration

Main idea: Set aside a small amount of time for exploration.

ε-Greedy Policy π^ε

The policy $\pi^\varepsilon(s)$ is defined as:

$$\pi^\varepsilon(s) = \begin{cases} \pi(s) & \text{with probability } 1-\varepsilon \\ \text{(uniform) random action,} & \text{with probability } \varepsilon \end{cases}$$

Algorithm revisited - Monte Carlo policy evaluation

Main idea: Set aside a small amount of time for exploration by using π^{ε}.

1. **Input**: policy π
2. **Initialize**:
   - $V(s) = 0$ for all $s \in S$
   - $N(s) = 0$ for all $s \in S$ (number of visits to s)
3. **Repeat**:
   - Generate a trajectory $s_0, a_0, r_0, s_1, a_1, r_1, \ldots s_T, a_T, r_T$ using π^{ε}
   - $G = 0$
   - For all time steps $t = T, T-1, \ldots, 0$:
     - $G = r_t + \gamma G$
     - $N(s_t) = N(s_t) + 1$
     - $V(s_t) = V(s_t) + \frac{1}{N(s_t)}(G - V(s_t))$

Monte Carlo for control

Same idea as policy iteration, but using π^{ε}. The evaluation part is made using Monte Carlo.

1. **Initialize**
   - $Q(s, a) \leftarrow 0$ for all $s \in S, a \in A$
   - $\pi(s) \leftarrow \arg\max_a Q(s, a)$ for all $s \in S$
2. **Repeat**:
   - Generate a trajectory $s_0, a_0, r_0, s_1, a_1, r_1, \ldots s_T, a_T, r_T$ using π^{ε}
   - $G = 0$
   - For all time steps $t = T, T-1, \ldots, 0$:
     - $G = r_t + \gamma G$
     - $N(s_t, a_t) = N(s_t, a_t) + 1$
     - $Q(s_t, a_t) = Q(s_t, a_t) + \frac{1}{N(s_t, a_t)}(G - Q(s_t, a_t))$
   - **Update the policy.** $\pi(s) = \arg\max_a Q(s, a)$

Bootstrapping

Monte Carlo methods caveats

What if trajectories are too long? Or never ends? (online setting)

- $\rightarrow$ use bootstrapping: estimating using estimates.

$$\begin{aligned} G_t &= r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots \\ &\approx r_t + \gamma \widehat{V}(s_{t+1}) \end{aligned}$$