MDPs: Q-learning

Q-learning

Problem: model-free Monte Carlo and SARSA only estimate Q_π, but want Q_{opt} to act optimally

Output	**MDP**	**reinforcement learning**
Q_π	policy evaluation	model-free Monte Carlo, SARSA
Q_{opt}	value iteration	**Q-learning**

- Recall our goal is to get an optimal policy, which means estimating Q_{opt}.
- The situation is as follows: Our two methods (model-free Monte Carlo and SARSA) are model-free, but only produce estimates Q_π. We have one algorithm, model-based value iteration, which can be used to produce estimates of Q_{opt}, but is model-based. Can we get an estimate of Q_{opt} in a model-free manner?
- The answer is yes, and Q-learning is an algorithm that accomplishes this.
- One can draw an analogy between reinforcement learning algorithms and the classic MDP algorithms. MDP algorithms are offline, RL algorithms are online. In both cases, algorithms either output the Q-values for a fixed policy or the optimal Q-values.

Q-learning

Bellman optimality equation:

$$Q_{\text{opt}}(s,a) = \sum_{s'} T(s,a,s')[\text{Reward}(s,a,s') + \gamma V_{\text{opt}}(s')]$$

Algorithm: Q-learning [Watkins/Dayan, 1992]

On each (s,a,r,s'):

$$\hat{Q}_{\text{opt}}(s,a) \leftarrow (1-\eta)\underbrace{\hat{Q}_{\text{opt}}(s,a)}_{\text{prediction}} + \eta\underbrace{(r + \gamma \hat{V}_{\text{opt}}(s'))}_{\text{target}}$$

Recall: $\hat{V}_{\text{opt}}(s') = \max_{a' \in \text{Actions}(s')} \hat{Q}_{\text{opt}}(s',a')$

- To derive Q-learning, it is instructive to look back at the Bellman optimality equation for Q_{opt}. There are several changes that take us from this recurrence to Q-learning. First, we don't have an expectation over s', but only have one sample s'.
- Second, because of this, we don't want to just replace $\hat{Q}_{\text{opt}}(s,a)$ with the target value, but want to interpolate between the old value (prediction) and the new value (target).
- Third, we replace the actual reward $\text{Reward}(s,a,s')$ with the observed reward r (when the reward function is deterministic, the two are the same).
- Finally, we replace $V_{\text{opt}}(s')$ with our current estimate $\hat{V}_{\text{opt}}(s')$.
- Importantly, the estimated optimal value $\hat{V}_{\text{opt}}(s')$ involves a maximum over actions rather than taking the action of the policy. This max over a' rather than taking the a' based on the current policy is the principle difference between Q-learning and SARSA.

SARSA versus Q-learning

Algorithm: SARSA

On each (s, a, r, s', a'):

$$\hat{Q}_\pi(s, a) \leftarrow (1 - \eta)\hat{Q}_\pi(s, a) + \eta(r + \gamma \hat{Q}_\pi(s', a'))$$

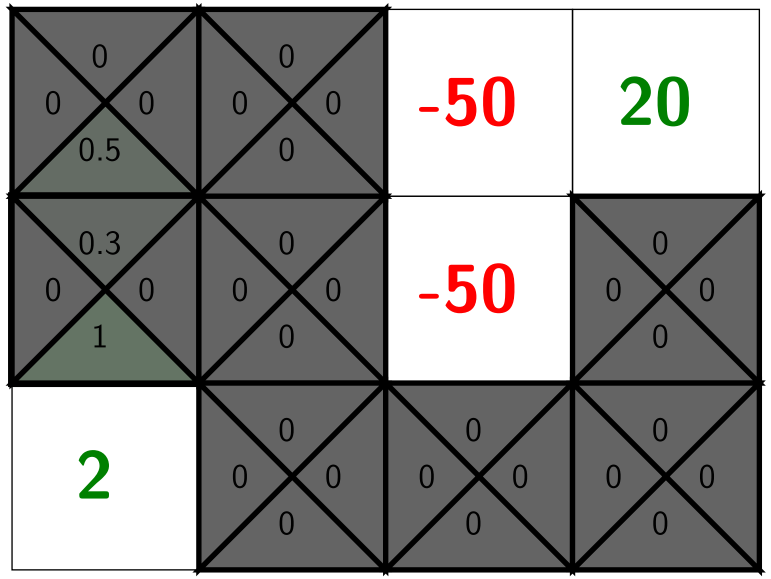
Algorithm: Q-learning [Watkins/Dayan, 1992]

On each (s, a, r, s'):

$$\hat{Q}_{\text{opt}}(s, a) \leftarrow (1 - \eta)\hat{Q}_{\text{opt}}(s, a) + \eta(r + \gamma \max_{a' \in \text{Actions}(s')} \hat{Q}_{\text{opt}}(s', a'))]$$

Volcanic SARSA and Q-learning

Run (or press ctrl-enter)



a	r	s
		(2,1)
N	0	(1,1)
S	0	(2,1)
S	2	(3,1)

Utility: 2

- Let us try SARSA and Q-learning on the volcanic example.
- If you increase `numEpisodes` to 1000, SARSA will behave very much like model-free Monte Carlo, computing the value of the random policy.
- However, note that Q-learning is computing an estimate of $Q_{\text{opt}}(s, a)$, so the resulting Q-values will be very different. The average utility will not change since we are still following and being evaluated on the same random policy. This is an important point for **off-policy** methods: the online performance (average utility) is generally a lot worse and not representative of what the model has learned, which is captured in the estimated Q-values.

Off-Policy versus On-Policy

Definition: on-policy versus off-policy

On-policy: evaluate or improve the data-generating policy

Off-policy: evaluate or learn using data from another policy

	on-policy	**off-policy**
policy evaluation	Monte Carlo SARSA	
policy optimization		Q-learning

- What do we mean by off-policy?
- Model-free Monte Carlo depends strongly on the policy π that is followed; after all it's computing Q_π. Because the value being computed is dependent on the policy used to generate the data, we call this an **on-policy** algorithm. In contrast, model-based value iteration is **off-policy**, because the model we estimated did not depend on the exact policy (as long as it was able to explore all (s, a) pairs).
- Further, model-free Q-learning is also **off-policy**, since it can learn the optimal policy using data from other policies.

Reinforcement Learning Algorithms

Algorithm	**Estimating**	**Based on**
Model-Based Monte Carlo	$\hat{T}, \hat{R}$	$s_0, a_1, r_1, s_1, \dots$
Model-Free Monte Carlo	$\hat{Q}_\pi$	u
SARSA	$\hat{Q}_\pi$	$r + \hat{Q}_\pi$
Q-Learning	$\hat{Q}_{\text{opt}}$	$r + \hat{Q}_{\text{opt}}$