Prof. Joschka Boedecker, Gabriel Kalweit, Maria Huegle, Andreas Saelinger

Reinforcement Learning
Exercise 4

1 Monte Carlo Prediction

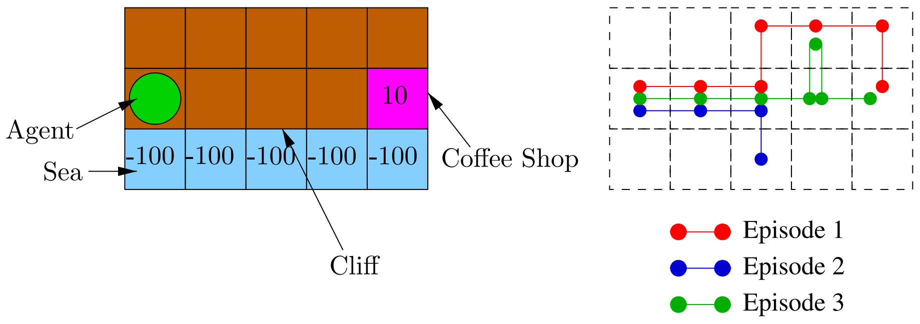

Figure 1: Cliff MDP

Consider the MDP in Figure 1, where all actions (an action moves the agent in a desired direction: up, down, left or right) succeed with a probability of 0.8. With a probability of 0.2 the agent moves randomly in another direction. All transitions result in a reward of -1, except when the coffee shop is reached (terminal state $s_{2,5}$: reward of 10) or if the agent falls of the cliff (terminal states $s_{3,1} \dots s_{3,5}$: reward of -100). The agent always starts in state $s_{2,1}$ as indicated in Figure 1.

Using Monte-Carlo policy evaluation, calculate $V(i)$ for all states i based on the illustrated episodes 1 to 3 (right part of Figure 1). Use the first-visit-method, i.e. every state is updated only once – on the first-visit – per episode, even if the state is visited again during the episode. In this task, we estimate the value by a running mean with $\alpha_t = \frac{1}{t}$ for episode t and initialize $V(i) = 0$ for all i. We do not discount, i.e. $\gamma = 1$.

2 Off-Policy MC Control with Importance Sampling

This task is based on the Blackjack example from the lecture[1] and an implementation can be found in `blackjack.py`. The state is a tuple – containing the players current sum, the dealer's one showing card (1-10 where 1 is ace) and whether or not the player holds a usable ace (0 or 1) – and the value is a float. You find the tests in `exercise-04_test.py`. They expect an average return. Run them by

```
python exercise-04_test.py -v
```

[1] `www.incompleteideas.net/book/RLbook2018.pdf#page=115`

or by

```
python -m unittest exercise-04_test.py -v.
```

In addition, you also find a visualization script of the predicted value-functions for which you need `matplotlib`[2]. You can run it by

```
python visualization.py.
```

Implement Off-Policy MC Control as introduced in the lecture,

```
mc_control_importance_sampling(env, num_episodes, behavior_policy, discount_factor=1.0),
```

in `off_policy_mc.py`.

Please note that the Off-Policy MC Control algorithm from the lecture is based on *Weighted Importance Sampling*. Instead of estimating v_π by the empirical mean:

$$V(s) = \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)} G_t}{|\mathcal{T}(s)|},$$

Weighted Importance Sampling uses a weighted average:

$$V(s) = \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)} G_t}{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)}},$$

to reduce variance.

3 Experiences

Make a post in thread *Week 04: Monte Carlo Methods* in the forum[3], where you provide a brief summary of your experience with this exercise and the corresponding lecture.

[2]`https://matplotlib.org/users/installing.html`
[3]`https://ilias.uni-freiburg.de/goto.php?target=frm_1837317&client_id=unifreiburg`