

Reinforcement Learning Lab

Lesson 2: Monte Carlo Methods

Davide Corsi and Alberto Castellini

University of Verona
email: davide.corsi@univr.it

Academic Year 2022-23



UNIVERSITÀ
di **VERONA**

Dipartimento
di **INFORMATICA**

Environment Setup

The first step for the setup of the laboratory environment is to update the repository and load the **miniconda** environment.

- Update the repository of the lab:

```
cd RL-Lab  
git stash  
git pull  
git stash pop
```

- Activate the *miniconda* environment:

```
conda activate rl-lab
```

Safe Procedure

Always back up the previous lessons' solutions before executing the repository update.

Today Assignment

In today's lesson, we implement the **On Policy Monte Carlo Control** algorithm in Python. In particular, the file to complete is:

`RL-Lab/lessons/lesson_2_code.py`

Inside the file, a function is partially implemented. The objective of this lesson is to complete it.

- **`def on_policy_mc()`**

Expected results can be found in the:

`RL-Lab/results/lesson_2_results.txt`

Pseudocode - On Policy Monte Carlo

On-policy first-visit MC control (for ε -soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

$\pi \leftarrow$ an arbitrary ε -soft policy

$Q(s, a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s, a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following π : $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \dots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1, \dots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

$A^* \leftarrow \operatorname{argmax}_a Q(S_t, a)$ (with ties broken arbitrarily)

For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$

Figure: Pseudocode for the on-policy monte carlo control algorithm, the implementation is from the Sutton and Barto book *Reinforcement Learning: An Introduction*

Assignment Notes

Today's assignment is based on the same environment as the first lesson (*DangerousGridWorld*). The suggested assignment's solution uses the `sample_episode()` function. Consult the first tutorial for more information.

First Visit vs Every Visit

The given pseudocode is for the first visit version. However, the most straightforward every-visit approach works for the *DangerousGridWorld* environment. The suggestion is to use the every visit approach, which does not require the check: *unless the pair S_t, A_t appears in ...* (6th line of pseudocode).

Results Disclaimer

Given the (high) stochasticity of the method, the obtained results may differ from those suggested. The crucial requirement is to obtain a policy that reaches the goal position.