Reinforcement Learning Lab

Lesson 4: Monte Carlo Reinforcement Learning Methods

Luca Marzari and Alberto Castellini

University of Verona
email: luca.marzari@univr.it

Academic Year 2023-24

Environment Setup

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

Safe Procedure

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

- Update the repository of the lab:

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

- Activate the *miniconda* environment:

```
conda activate rl-lab
```

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_4_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:

```
RL-Lab/results/lesson_4_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:

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

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

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

Repeat forever (for each episode):

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

$\quad G \leftarrow 0$

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

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

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

$\quad\quad\quad$ Append G to $Returns(S_t, A_t)$

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

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

$\quad\quad\quad$ 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 ... (6^{th} 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.