Online and Reinforcement Learning

2024-2025

Home Assignment 8

Sadegh Talebi
Department of Computer Science
University of Copenhagen

The deadline for this assignment is **11 April 2025, 23:59**. You must submit your *individual* solution electronically via the Absalon home page.

A solution consists of:

- A PDF file with detailed answers to the questions, which may include graphs and tables if needed. Do *not* include your full source code in the PDF file, only selected lines if you are asked to do so.
- A .zip file with all your solution source code with comments about the major steps involved in each question (see below). Source code must be submitted in the original file format, not as PDF. The programming language of the course is Python.

Important Remarks:

- IMPORTANT: Do NOT zip the PDF file, since zipped files cannot be opened in *SpeedGrader*. Zipped PDF submissions will not be graded.
- Your PDF report should be self-sufficient. I.e., it should be possible to grade it without opening the .zip file. We do not guarantee opening the .zip file when grading.
- Your code should be structured such that there is one main file (or one main file per question) that we can run to reproduce all the results presented in your report. This main file can, if you like, call other files with functions, classes, etc.
- Handwritten solutions will not be accepted.

1 MDP Classes `(20 points) [Sadegh]`

For each of the following MDPs, determine whether it belongs to the class of ergodic MDPs, communicating MDPs, or weakly-communicating MDPs. Then fill in Table 1 with `True` or `False`. (Note that all entries must be filled in.)

In each case, argue why membership to a certain class holds. To rule out membership, you must provide a concrete counter example; e.g., if M is not ergodic, you must provide a policy that violates the definition of ergodic MDP.

(i) RiverSwim (Figure 1).

(ii) A modified RiverSwim (let us call it RiverSwim-2), which is defined similarly to RiverSwim except that it has an additional state $s_{\texttt{extra}}$ with two actions a_1 and a_2: Under action a_1, the agent is moved to 1 (left-hand bank) deterministically; under action a_2, the next-state is 1 (w.p. 0.5) and 2 (w.p. 0.5). The rest of the transitions are the same as in RiverSwim.

MDP \ Class	Ergodic	Communicating	Weakly-Communicating
RiverSwim			
RiverSwim-2			
4-room grid-world			

Table 1: MDP Classes

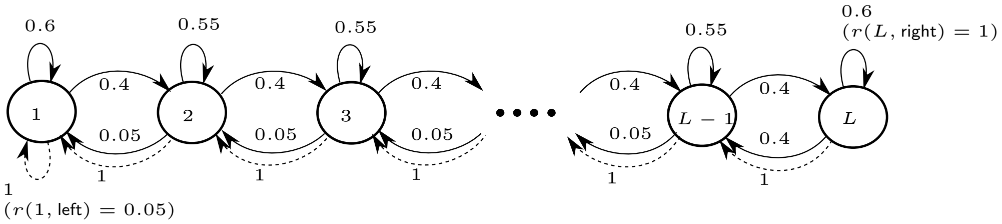

Figure 1: The RiverSwim MDP

(iii) 4-room grid-world, which was introduced in earlier assignments. We present it below for completeness. This MDP is shown in Figure 2. The agent has 4 actions (when away from walls): Going up, left, down, or right. However, the floor is slippery and brings stochasticity to the next-state. Specifically, under each of the four aforementioned actions, she moves in the chosen direction (with probability 0.7), stays in the same state (with probability 0.1), or goes in each of the two perpendicular directions (each with probability 0.1) —this environment is sometimes referred to as the *frozen lake* MDP. *Walls act as reflectors, i.e., they cause moving back to the* ***current state****.* Once the agent reaches the rewarding state (in yellow), she is *teleported* to the initial state irrespective of the action, that is, she will find herself in the upper-left corner at the beginning of the following slot.

2 Infinitely Many Choices for Optimal Bias in Average-Reward MDPs (15 points) [Sadegh]

Consider a finite weakly-communicating average-reward MDP $M = (\mathcal{S}, \mathcal{A}, P, R)$, and assume that $b^\star : \mathcal{S} \to \mathbb{R}$ is an optimal bias function for M; namely, it satisfies the optimal Bellman equations of M. Now

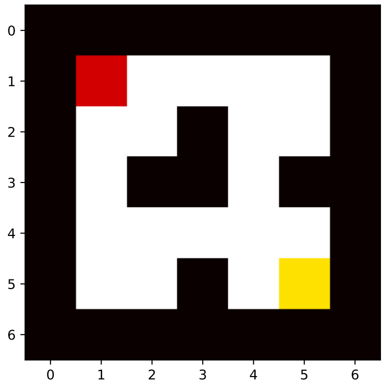

Figure 2: The 4-Room Grid-World MDP

consider a function $b' : \mathcal{S} \to \mathbb{R}$ that admits the following form:

$$\forall s \in \mathcal{S} : \quad b'(s) = b^\star(s) + c$$

for some constant $c \in \mathbb{R}$. Show that b' is also an optimal bias function for M for any $c \in \mathbb{R}$.

(Alternatively, one can view $b^\star$ as a vector living in $\mathbb{R}^S$, where S is the size of $\mathcal{S}$. Then, the question asserts that $b' = b^\star + c\mathbf{1}$ is also an optimal bias function, where $\mathbf{1}$ is a vector of all ones.)

3 Comparison between UCRL2 and UCRL2-L `(40 points)` `[Sadegh]`

The goal of this exercise is to perform an empirical evaluation of `UCRL2` (Jaksch et al., 2010) and `UCRL2-L` (see slides), and compare their regret. The original algorithm `UCRL2` differs from `UCRL2-L` only in the choice of confidence sets. `UCRL2` relies on the following confidence sets for P and R: For all $(s, a) \in \mathcal{S} \times \mathcal{A}$,

$$C_{s,a}^{\texttt{UCRL2}} = \left\{ R' \in [0,1] : |\widehat{R}_t(s,a) - R'| \le \sqrt{\frac{3.5}{N_t(s,a)} \log\Big(\frac{2SAt}{\delta}\Big)} \right\},$$

$$C_{s,a}^{\prime\texttt{UCRL2}} = \left\{ P' \in \Delta(\mathcal{S}) : \|\widehat{P}_t(\cdot|s,a) - P'\|_1 \le \sqrt{\frac{14S}{N_t(s,a)} \log\Big(\frac{2At}{\delta}\Big)} \right\}.$$

Those used in `UCRL2-L` are defined in the slides and an implementation of it is provided in the accompanied Python file `UCRL2_L.py`.

(i) Implement `UCRL2`. (You can directly modify `UCRL2_L.py`, but provide a code snippet to highlight necessary modifications.)

(ii) Examine both `UCRL2` and `UCRL2-L` on a 6-state RiverSwim (Figure 1). In the experiment, set the initial state to the left-most state (state 1), and set the time horizon $T = 3.5 \times 10^5$. As for failure probability δ, set: $\delta_{\texttt{UCRL2}} = 0.05$ and $\delta_{\texttt{UCRL2-L}} = \frac{1}{4} \times 0.05 = 0.0125$ —this adjustment ensures that the regret upper bounds of the two algorithms will have identical failure probabilities; thus, it offers a (more) fair comparison between the two algorithms.

We are interested in the following definition of regret, as implemented in the file:

$$\mathfrak{R}(T) = \sum_{t=1}^{T} (g^\star - r_t).$$

Report the empirical regret curves under both algorithms, averaged over 50 independent runs, and report the corresponding 95% confidence intervals. (Recall that for a set of i.i.d. random variables $\mathbf{X} = \{X_1, \dots, X_n\}$, we define the 95% confidence interval by

$$\Big[\texttt{mean}(\mathbf{X}) - 1.96\frac{\texttt{std}(\mathbf{X})}{\sqrt{n}}, \texttt{mean}(\mathbf{X}) + 1.96\frac{\texttt{std}(\mathbf{X})}{\sqrt{n}}\Big],$$

where `mean` and `std` denote the mean and the standard deviation, respectively.)

(iii) Plot the *empirical gain* $\frac{1}{t}\sum_{t'=1}^{t} r_t$ under each algorithm. In the plot, also add the horizontal line corresponding to the optimal gain $g^\star$.

(iv) Report the average number of episodes initiated by both algorithms.

(v) Using the results obtained in (ii)-(iv), discuss the differences between `UCRL2-L` and `UCRL2` (from an empirical standpoint).

4 Leveraging Prior Knowledge on the Graph of MDP to Reduce Regret (25 points) [Sadegh]

In this exercise, we study a variant of `UCRL2-L` that leverages prior knowledge to reduce exploration, and hence, regret. Specifically, we are interested in incorporating some prior knowledge on the transition graph of the MDP into the algorithm. For a given (s,a) pair, let $\mathcal{K}_{s,a}$ denote the *support set* of the distribution $P(\cdot|s,a)$:

$$\mathcal{K}_{s,a} := \Big\{x \in \mathcal{S} : P(x|s,a) > 0\Big\}.$$

In words, $\mathcal{K}_{s,a}$ is the set of all possible next-states when choosing action a in state s. For example, in RiverSwim, $\mathcal{K}_{2,\mathsf{left}} = \{1\}$ whereas $\mathcal{K}_{2,\mathsf{right}} = \{1,2,3\}$. In the general setting of RL, not only P but also its associated support sets $\mathcal{K}_{s,a}, s \in \mathcal{S}, a \in \mathcal{A}$ are assumed unknown to the agent. We are interested in studying an intermediate setting, where P is unknown, but the associated support sets are known through, e.g., some domain knowledge. Intuitively, this corresponds to knowing the transition graph of the MDP, but without knowing the actual probabilities.

Assume that support set $\mathcal{K}_{s,a}$ for each pair (s,a) is provided to the agent. The agent can then leverage this prior knowledge to *rule out* some candidate models in $\mathcal{M}_t$, the high-probability set of plausible models maintained by `UCRL2-L` at time t. (For example, in RiverSwim, the agent would know that a model that would include a next-state of 4 under $(2, \mathsf{right})$ is certainly wrong as it contradicts with the prior knowledge $\mathcal{K}_{2,\mathsf{right}}$; such a model could therefore be shaved off from $\mathcal{M}_t$.)

(i) Argue how the confidence set for P could be modified to incorporate prior knowledge of the support sets. Write down the precise mathematical form of the revised confidence sets.

Then modify `UCRL2-L` using this new confidence set. Let us call this algorithm `UCRL2-L-supp`. Implement `UCRL2-L-supp` (i.e., modify `UCRL2_L.py`), and examine it in the 6-state Ergodic RiverSwim (Figure 3). In the experiment, set the initial state to the left-most state, and set $T = 4 \times 10^5$ and $\delta = 0.05$. Report the empirical regret, averaged over 40 independent runs, along with the corresponding 95% confidence intervals.

(ii) Repeat the experiment in Part (i) for `UCRL2-L` (i.e., without prior knowledge) and discuss the results.

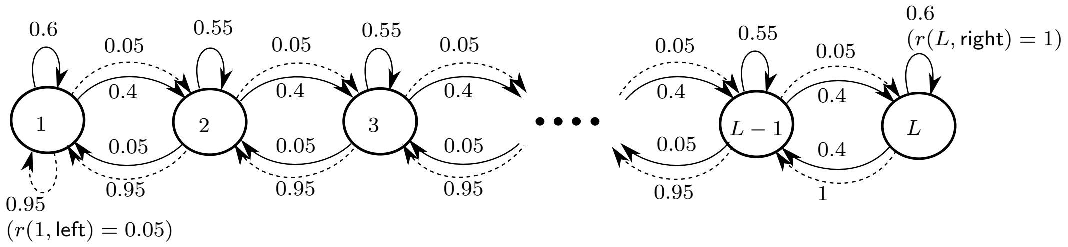

Figure 3: The L-state Ergodic RiverSwim MDP

(Optional) Further, we are interested in studying another intermediate setting where P is unknown, but **cardinality** of the associated support sets are known. Hence, the agent knows the *number* of possible next-states under various state-action pairs. Intuitively, this corresponds to knowing *the degree of various nodes* in the transition graph of the MDP, but without knowing the actual probabilities or the elements of $\mathcal{K}_{s,a}$.

(iii) (Optional) Now assume that $|\mathcal{K}_{s,a}|$ for each pair (s,a) is known a priori. Derive a variant of `UCRL2-L`, which we call `UCRL2-L-SuppSize`, that can use this prior knowledge. Implement `UCRL2-L-SuppSize` and examine it in 6-state Ergodic RiverSwim using the same setting as above. Report the empirical regret, averaged over 50 independent runs, along with the corresponding 95% confidence intervals. Compare the result with those of Parts (i)-(ii) above.

References

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11, 2010.