Reinforcement Learning Tutorial 9, Week 10
— with solutions —
Revision: MDPs & Semi-Gradient SARSA

Pavlos Andreadis

March 2025

Overview: The following tutorial questions relate to material taught in weeks 1 to 6 of the 2024-25 Reinforcement Learning course. They aim at encouraging engagement with the course material and facilitating a deeper understanding.

We continue this week with another look at past exam questions. The first one involves modelling a Markov Decision Process (MDP) for a variation of *the River Crossing Riddle* [**?**]. Note how the problem explicitly defines your state space, and does not allow for defining any further states. This constraints your choices when asked to define absorbing states in the first question. Note also that transitions here are deterministic.

In the second question, taken from the same exam, you are first asked to define a set of features for the state space. Though you are not allowed to use a one-hot encoding, there are still a few alternative formulations to explore. Some of them are preferable to others, so it is worth thinking about how your feature space affects what value function approximations can be learned.

Problem 1 - Revision: MDP Modelling

The River Crossing Riddle **[Adapted from RL exams in 2018-19]**

A farmer went to a market and purchased a wolf, a goose, and a bag of beans. On his way home, the farmer came to the bank of a river and rented a boat. But crossing the river by boat, the farmer could carry only himself and a single one of his purchases: the wolf, the goose, or the bag of beans. If left unattended together, the wolf would eat the goose, or the goose would eat the beans. The farmer could use the boat to transfer his purchases from either bank of the river to the other. The farmer could also move across the river with the boat empty.

Consider the control problem for moving all of the farmer's purchases across the river, where the current state is defined *only* by the following binary state variables:

- which side of the river the farmer is at;
- which side of the river the wolf is at;
- which side of the river the goose is at; and
- which side of the river the bag of beans is at.

1. Define a state space for the control problem. Which is the initial state? Which states need to be defined as absorbing states?

   Answer:

   - Define 2^4 different states. For example: s_{wgbf} with $w, g, b, f \in 0, 1$, where w, g, b, f represent the 4 state variables in order, and 0 puts the respective item at the starting bank, while 1 puts it across the river. (Following this, only s_{1110} is an unreachable state).
   - Following the above, s_{0000} is the initial state.
   - Absorbing states: The goal state s_{1111} and any state where $w = g \neq f$ and/or $g = b \neq f$ (these are s_{0001}, s_{1001}, s_{0011}, s_{0110}, s_{1100}).

2. Given the above information, formulate a finite Markov Decision Process (MDP) with discounting for the problem of controlling the transfer of the farmer's purchases across the river. For the transition function, give only transitions from the initial state.

   Answer:

   - Actions can be defined in a variety of ways, but the easiest way is to define 4 actions: one action for transporting each purchase, and one for moving across the river with no purchase in the boat. The last action is always available, and the other 3 only when the respective purchase is on the same side of the river as the farmer.
   - Any reward function that maps a reward to arriving at every absorbing state, with the reward for the goal state being the largest. Any difference between the non-goal absorbing states should be rationalised (e.g. the goose is "pricier"). A penalty for every non-terminating action is reasonable, as long as the non-goal absorbing states have a considerably larger penalty or the goal state a considerably larger reward.
   - The transition function is deterministic. For the state $s = s_{0000}$, $T(s_{0001} \mid s, a_0) = T(s_{1001}|s, a_1) = T(s_{0101}|s, a_2) = T(s_{0011}|s, a_3) = 1$.
   - All that's left is to define γ. Any value in $(0, 1]$ is acceptable.

Problem 2 - Revision: Semi-Gradient SARSA

[Adapted from RL exams in 2018-19]

You observe an agent going through the following sequence of states, actions (i.e. `move-up`, `move-left`), and reward signals.

$$\{\text{“green”, “loud”}\}, \texttt{move-up}, -1$$
$$\{\text{“blue”, “loud”}\}, \texttt{move-left}, 0$$
$$\{\text{“green”, “silent”}\}, \texttt{move-up}$$

1. Define a set of features to represent the state (do not enumerate the states; in other words: do not define one feature per state, and do not use a state index as a feature). For simplicity, assume that states are only ever described as either “green” or “blue”, and as either “loud” or “silent”, and that the only available actions are `move-up` and `move-left`.

   ## Answer:

   A good setup is 1 binary feature per keyword. Less good is 2 binary features, on colour and loudness, unless a feature for the bias is also added (with an assignment of 1, always) which would again make this a good setup. You may or may not include features for the actions as part of this vector. Two viable answers, which the answer below assumes, would be:

   (a) $[x_{\text{green}}, x_{\text{blue}}, x_{\text{loud}}, x_{\text{silent}}]$

   (b) $[x_{\text{green}}, x_{\text{blue}}, x_{\text{loud}}, x_{\text{silent}}, x_{\text{up}}, x_{\text{left}}]$,

   where all variables should have been defined as binary.

2. Then, using linear function approximation for the action-value function in this problem compute the first 2 semi-gradient one-step Sarsa updates along the episode and given the actions taken, using initial weights of 0, a discount factor of 1, and a learning rate of 0.1 (show all the steps).

Answer:

Example solutions, for the above examples of feature definitions:

(a) $w_0^{up} = [w_0^{\text{green},up},\ w_0^{\text{blue},up},\ w_0^{\text{loud},up},\ w_0^{\text{silent},up}] = [0,0,0,0]$
$w_0^{left} = [w_0^{\text{green},left},\ w_0^{\text{blue},left},\ w_0^{\text{loud},left},\ w_0^{\text{silent},left}] = [0,0,0,0]$
$\alpha = 0.1$, $\gamma = 1$.

- Step 1:

$$\begin{aligned} w_1^{up} &= w_0^{up} && +\alpha && [R_0 + \gamma \cdot w_0^{left\top} x_1 && -w_0^{up\top} x_0 &&] \cdot x_0 \\ &= [0,0,0,0] && +0.1 && [-1 + 1 \cdot 0 && -0 &&] \cdot [1,0,1,0] \\ &= -0.1 \cdot [1,0,1,0] && . \end{aligned}$$

- Step 2: ($w_1^{left} = w_0^{left}$)

$$\begin{aligned} w_2^{left} &= w_1^{left} && +\alpha && [R_1 + \gamma \cdot w_1^{up\top} x_2 - w_1^{left\top} x_1] \cdot x_1 \\ &= [0,0,0,0] && +0.1 && [0 + 1 \cdot [-0.1,0,-0.1,0]^\top [1,0,0,1] - 0] \cdot [0,1,1,0] \\ &= -0.01 \cdot [0,1,1,0] \end{aligned}$$

(b) $w_0 = [w_0^{\text{green}},\ w_0^{\text{blue}},\ w_0^{\text{loud}},\ w_0^{\text{silent}},\ w_0^{\text{up}},\ w_0^{\text{left}}] = [0,0,0,0,0,0]$
$\alpha = 0.1$, $\gamma = 1$.

- Step 1:

$$\begin{aligned} w_1 &= w_0 && +\alpha && [R_0 + \gamma \cdot w_0^\top x_1 && -w_0^\top x_0 &&] \cdot x_0 \\ &= [0,0,0,0] && +0.1 && [-1 + 1 \cdot 0 && -0 &&] \cdot [1,0,1,0,1,0] \\ &= -0.1 \cdot [1,0,1,0,1,0] \end{aligned}$$

- Step 2:

$$\begin{aligned} w_2 &= w_1 + \alpha [R_1 + \gamma \cdot w_1^\top x_2 - w_1^\top x_1] \cdot x_1 \\ &= [-0.1,0,-0.1,0,-0.1,0] + 0.1 \Big[0 + \\ &\quad + 1 \cdot [-0.1,0,-0.1,0,-0.1,0]^\top [1,0,0,1,1,0] \\ &\quad - [-0.1,0,-0.1,0,-0.1,0]^\top [0,1,1,0,0,1] \Big] \cdot [0,1,1,0,0,1] \\ &= [-0.1,0,-0.1,0,-0.1,0] - 0.01 \cdot [0,1,1,0,0,1] \\ &= [-0.1,-0.01,-0.11,0,-0.1,-0.01]. \end{aligned}$$