Reinforcement Learning Tutorial 2, Week 3
— with solutions —
Multi-Armed Bandits / MDPs

Pavlos Andreadis, Sanjay Rakshit, Adam Jelley

January 2024

Overview: The following tutorial questions relate to material taught in the first two weeks of the 2023-24 Reinforcement Learning course. They aim at encouraging engagement with the course material and facilitating a deeper understanding.

In this week of tutorials we will start looking into modelling "real-world" problems as Reinforcement Learning problems. I quote "real-world" as we will typically resort to what is often called a "toy-problem" in academia. Part of this is because they are easier to write, and much easier to limit in scope. Part of it is because it allows us to poke controllable holes into the problems and make the limitations and assumptions that much clearer. Rather than overwhelming you with potential considerations (and more realistic problems especially from your everyday experience would trigger many such personalised thoughts that we can't always predict), you are introduced to them one at a time. It is much easier for you to focus on what we are trying to communicate when you accept from the start that it is a "story", and not exactly reality.

The descriptions of the problems will still not always be a perfect specification; that is after all what is required of you to produce when modelling it. The description will however hint at the correct level of abstraction. For example, in these problems we are presenting a frog climbing onto a rock as if it were a trivial matter, and do not give any detailed information on that process.

[*Could be fun to consider what such information might be. By the way, you will see a lot of italicised brackets like these, especially in the solutions. These are just here for you to contemplate further, or perhaps discuss with your colleagues or tutor if you so want.*]

Problem 1 - Modelling: Frog on a Rock

A friendly frog, Hop Along was stranded on a rock surrounded by water. It needs to get to land without falling in. The only way to safety is for it to jump on to neighbouring rocks till it arrives on land.

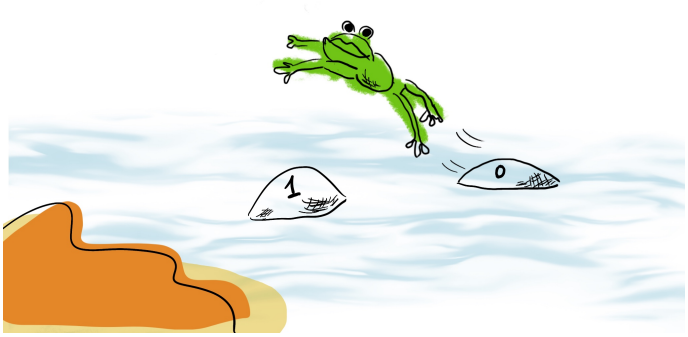


Figure 1: "Will He Make it?" (Image and title from Yana Knight [2021], with permission).

There are two rocks, and Hop Along can jump from a rock to another or from the final rock to land. However, sometimes it misses and ends up in the water, having to climb back onto the same rock it tried to jump from. The rocks are arrayed in a row leading from its starting rock, which we will name $rock_0$, to $rock_1$, and then finally land which can only be reached from $rock_1$.

Consider the control problem where the current state is specified by Hop Along's location and the actions Hop Along can take is to attempt to jump towards another reachable rock or land.

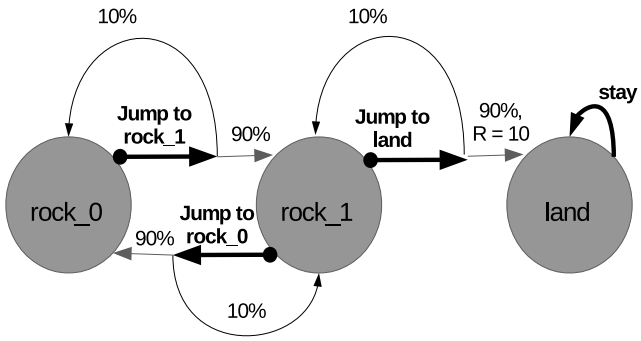
Assume that Hop Along's jumps always have a 90% chance of reaching the intended destination, while the rest of the time Hop Along falls in the water.

Formulate a Markov Decision Process (MDP, see Sutton and Barto [2018], Ch. 3) for the problem of deciding on Hop Along's actions in order to help it reach land. Why did you formulate it as you did? What additional assumptions did you have to make?

Can you define the uniform random policy on this MDP? What about the optimal policy?

Answer: We define three states, $rock_0$, $rock_1$, and *land*. We will assume that the episode terminates once we have reached land, as that is Hop Along's goal.

For the same reason, a reasonable reward function could give a reward of 10 for reaching land (with all other rewards 0). Though other values could be used.



Graphical model for Problem 1.

The state *land* will be an absorbing state as we are not interested in further controlling Hop Along after that. We can reach *land* from all our potential starting states, as long as the Reinforcement Algorithm we use does not get caught in a loop between $rock_0$ and $rock_1$. If we can guarantee that, then we can set our discount factor to 1.

[*Can you see how such a loop might happen?*]

We could set a smaller discount factor (e.g. 0.9) as this will help prioritise policies that reach *land* sooner. Another way of achieving the latter would be to add a small negative reward to, either all transitions to $rock_0$ and $rock_1$, or all actions that are not *"Jump to land"*, as the negative rewards would accumulate the longer the task takes to complete.

As *land* is an absorbing state, we have to define one action from it (which we have named *"stay"*) that 100% transitions the state back to *land*, and with a reward of 0.

For defining the policies, the uniformly random policy is defined for each state such that the probability of each possible action from that state is equal. In this case, $\pi(a_1|rock_0) = 1$, $\pi(a_{land}|rock_1) = 0.5$ and $\pi(a_0|rock_1) = 0.5$. We may also consider the possibility of a 'no-op' action where Hop Along stays at the current location. This would of course change the possible actions and the corresponding probabilities.

The optimal policy, however, depends on the reward we receive for taking each action (and there may be multiple optimum policies depending on the reward model), so we cannot define (or learn) this until the reward model has been

defined. In this case, assuming we receive a reward of 10 for reaching land and 0 otherwise, and our discount factor is 0.9, then we do have a unique, deterministic optimal policy: $\pi(a_1|rock_0) = 1$, $\pi(a_{land}|rock_1) = 1$. However, note that if the discount factor is 1 (there is no penalty for falling in the water), the optimal policy is not unique; in fact there are an infinite number of stochastic optimal policies with different distributions for $\pi(a|rock_1)$ (the random policy included) since they will all have a return of 10 eventually!

Problem 2 - Modelling: Frog on a Rock 2

Though in Problem 1 we modelled our frog as always getting back on the rock it falls off, reality does not always comply with our assumptions (or conform to our expectations) and Hop Along the frog ended up being carried away by the river. It now resides in a calm little pond, with a nice rock in the middle, and surrounded by flies. It spends its time climbing onto this rock, and then jumping into one of the four cardinal directions (South, West, North, East) and swallowing the flies it comes across on its way down to the water.

Figure 2: "Frog with Problems." (Image and title from Yana Knight [2021], with permission).

Part a

Formulate Hop Along's attempt to catch as many flies as possible as a Multi-Armed Bandit (MAB) problem (see Sutton and Barto [2018], Ch. 2).

Answer: We have four actions/arms to choose from at each time-step: $\{South, West, North, East\}$. In each time-step, we select one of the four actions and receive a reward (in number of flies) from the environment. Our goal is to maximise the expected total number of flies across time.

Part b

Can you define two simple exploration strategies that Hop Along could employ to try to maximise the total number of flies he catches? What is an advantage and disadvantage of each potential strategy?

Answer: Hop Along could employ ϵ-greedy or UCB (Upper Confidence Bound) strategies, for example (see lecture 2 for details). UCB is likely to give a lower *regret* (defined after T rounds as the difference between the total number of flies caught, and the expected total number of flies that would have been caught, if the optimal action had been taken from the beginning), if we can assume a well-behaved reward distribution (e.g. Gaussian), but ϵ-greedy action selection will be simpler and does not require an assumption on the reward distribution. There are of course other possible strategies with their own assumptions and corresponding advantages and disadvantages.

Part c

An implied assumption in **Part a** is that the frog will never stop jumping. If there was a limited amount ϕ of jumps the frog could do (let's say $\phi = 100$ jumps), would it still make sense to model this problem as a MAB Problem? Why? If not, how would you go about modelling this scenario?

Answer: The problem with a limited number of jumps is that we can no longer assume that we are taking actions to infinity. In other words, we can only take a limited number of actions. [*Of course, there might be values of ϕ for which we can still practically assume taking actions to infinity. Can you tell why?*]

(You can also imagine a variation of this problem where the budget is on the available number of flies. Then we are still limited in the number of jumps, though we might not know exactly how many that is before running the experiment. [*Why don't we know exactly how many actions we can take in this version?*])

It would make sense to instead model this limited number of jumps scenario as a Reinforcement Learning (RL) problem that includes a state variable for the number of jumps left/taken. As is, we would have to look for model-free RL algorithms to solve that problem. [*What part of the environment model would we not have access to?*]

Ultimately however, even though some theoretical guarantees do not hold when we have to take a finite set of actions [*Which? (see lecture 2 slides)*], we can

still model our problem as a MAB. There is no explicit assumption of an infinite horizon (i.e. end time).

Note, that there are alternative MAB formulations in the literature which could be used here instead, such as *"Budget-Limited Multi-Armed Bandits"* Tran-Thanh et al. [2010]; This is an entirely optional and not necessarily useful reading, other than to point out that there are variants to the problem in the literature. However, it has been asked in class if there are cases where we treat the exploration as a different phase from exploitation, and this paper provides one such scenario, if you are curious.

You could also more generally look into MAB solutions that explicitly address the finite horizon (limited number of actions if you consider that we take one action per time-step). Try *"Finite-Horizon Multi-Armed Bandit"* (with and without quotations) in your favourite search engine (the search results will not be as clean as you might want them, but this is often the case with such searches).

Part d

Let's go back to assuming an infinite number of jumps. Another implied assumption in **Part a** has been that we can control Hop Along's actions. If we couldn't though (perhaps because we are not Hop Along ourselves, but a researcher interested in how many flies it eats), would it still make sense to model this as a MAB problem? Why? If not, how would you go about modelling this scenario?

Answer: The apparent answer is "no". A Multi-Armed Bandit (MAB) problem assumes access to a space of actions[1] at each time-step. If there is no action to take or, in other words, no decision to make, then we cannot formulate a MAB problem.

It might be best to simply model this as a statistical inference problem, where we are trying to learn a distribution over Hop Along's behaviour in relation to its jumps so far.

One can of course still imagine formulating the problem as a MAB from the frog's perspective and then comparing the frog's actual behaviour to the solutions we come up with (perhaps we are interested in seeing whether frog behaviour approximates ϵ-greedy behaviour).

[1] or specifically a discrete set of actions in the examples we have seen; for a continuous space of actions or a set of actions with similarity between actions, you could look into Magureanu et al. [2014].

Finally, to answer the question: *What if the much-troubled frog chasing flies was an oil-rig company sampling oil deposits, and the four cardinal directions were four different extraction sites?*, we would need to discuss the cost of the test digs as well as the general question of using of non-renewable energies or the dangers of oil spills for marine life, which all suggests a wider study of the literature and of the problem than possible within this course. Nevertheless, the methods that are taught in this course can be useful also in more complex situation when we cooperate with the right experts and stakeholders.

References

Yana Knight. "Story of Yana". http://storyofyana.com/, 2021.

Stefan Magureanu, Richard Combes, and Alexandre Proutiere. Lipschitz bandits: Regret lower bounds and optimal algorithms. *arXiv preprint arXiv:1405.4758*, 2014.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Long Tran-Thanh, Archie Chapman, Jose Enrique Munoz De Cote Flores Luna, Alex Rogers, and Nicholas R Jennings. Epsilon–first policies for budget–limited multi-armed bandits. *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*, 2010.