ASEN 5519-003 Decision Making under Uncertainty
Homework 4: Tabular Reinforcement Learning

February 27, 2024

1 Conceptual Questions

Question 1. (30 pts) Consider a 3-armed Bernoulli Bandit with payoff probabilities $\theta = [0.2, 0.3, 0.7]$.

a) After a very large number of pulls, what is the expected payoff per pull of an ϵ-greedy policy with $\epsilon = 0.15$ (and no decay)?
b) After a very large number of pulls, what is the probability of selecting arm 3 when using a softmax policy with $\lambda = 5$ (and a "precision factor" of 1.0)?
c) Suppose that you are maintaining a Bayesian belief over the parameters θ starting with initial prior of Beta(1,1). Plot or sketch[1] the pdfs of the posterior probability distributions for each θ assuming the following numbers of wins and losses for each arm: $w = [0, 1, 3]$, $l = [1, 0, 2]$.
d) Given the situation in (c), describe one iteration of Thompson sampling. What quantities are sampled from what distributions? Choose some plausible values for the random samples and indicate which arm will be pulled.

Question 2. (20 pts) Consider the following simple MDP: $S = \{1, 2\}$, $A = \{L, R\}$. The initial state is 1, and 2 is a terminal state. Both actions result in deterministic transitions to state 2. $R(1, L) = 10$, $R(1, R) = 20$. Consider a policy parameterized with $\theta = [\theta_L, \theta_R]$, where

$$\pi_\theta(a \mid s) = \frac{e^{\theta_a}}{e^{\theta_L} + e^{\theta_R}}.$$

Calculate the policy gradients at $\theta = [0.5, 0.5]$ without baseline subtraction for two trajectories: $(1, L, 10, 2)$ and $(1, R, 20, 2)$.

2 Exercises

Question 3. (50 pts) Implement **two** tabular or deep learning algorithms to learn a policy for the `DMUStudent.HW4.gw` grid world environment. You will submit the following deliverables:

a) The **source code** for both of the algorithms.

b) **Two learning curve plots**[2]. The y-axis of both plots should be the average **undiscounted** reward per episode from the *learned policy* (not the exploration policy). Each plot should contain learning curves from both algorithms for easy comparison. The x-axis should be as follows for the two plots, respectively:

   1) The number of steps taken in the environment (calls to `act!`).

[1] You may wish to use `https://homepage.divms.uiowa.edu/~mbognar/applets/beta.html` for this.
[2] These are the same plots shown in the SARSA notebook from class, and you may copy code from there.

2) The cumulative wall-clock time for training.

c) **Write** a short paragraph describing the relative strengths of the algorithms. Which one has higher sample complexity? Which one learns faster in terms of wall clock time?

Some algorithms to consider implementing are:

- Policy Gradient
- Max-Likelihood Model Based RL
- Q-Learning[3]
- SARSA
- Actor Critic

Please meet the following requirements and consider the following tips:

1. *One* of algorithms may be copied from the course notebooks or from any other reinforcement learning library you can find online, but at least one must be implemented by you from scratch or modified from the notebooks. You may also implement both from scratch.
2. It is possible to achieve average **undiscounted** cumulative reward per episode of greater than 5. At least one of your algorithms must reach this level of performance.
3. Use only the functions from `CommonRLInterface` to interact with the environment, and use the `HW4.render` function if you want to render the environment.
4. You may also wish to modify these algorithms with techniques discussed in class, such as improved exploration policies, eligibility traces, double Q learning, or entropy regularization.

[3]Q-Learning is probably the easiest of these to implement since it is only a small modification from SARSA.