COMP5600/6600: HW3

November 10, 2018

Requirement

Please submit your homework on Canvas before the deadline. If you need to use your delay coupon, you can hand in your homework to my office. Include your name, the student ID number, and how many days late it is (if handed in late) in the headline. If you have any questions, please contact our TA. (*), (**), or (***) indicates the difficulty of each question.

Policy

We apply the late policy explained in syllabus to all homework. Any grading questions must be raised with the TA in one week after the homework is returned. The homework must be completed individually. However, you are encouraged to discuss the general algorithms and ideas with classmates in order to help you answer the questions. You are also allowed to share examples that are not on the homework in order to demonstrate how to solve problems. If you work with one or more other people on the general discussion of the assignment questions, please record their names over every question they participated. However, the following behaviors will receive heavy penalties (lose all points and apply the honest policy explained in syllabus)

- explicitly tell somebody else the answers;
- explicitly copy answers or code fragments from anyone or anywhere;
- allow your answers to be copied;
- get code from Web.

Programming assignment

You can use any programming language you feel comfortable. The code is required to submit as an attachment. It is your obligation to provide runnable code which generates the result you submit. The fail to do that will cause losing points in this question. You should implement your own algorithm. Directly calling existing functions for this algorithm from any programming languages is not permitted, but you are allowed to reuse any existing functions to plot and print out your figure.

1 Shortest Path in a Graph as an MDP (5 points)

Consider the problem of finding the shortest path in a graph (V, E) (with V the set of vertices, and E the set of directed edges), and where vG is the destination vertex. Every edge in the graph takes equally long to traverse. Describe how this problem can be encoded into a Markov decision process, (S, A, T, γ, R), such that the optimal solution in the Markov decision process is the shortest path in the graph. Your MDP is required to have rewards that are positive (zero included) for all transitions.

2 Plotting recency-weighted averages (10 points)

Equation 2.5 (link) is a key update rule we will use throughout the course. This exercise will give you a better hands-on feel for how it works. For this exercise we'll be considering recency-weighted averages of a target signal from time t=0 through time t=15. For all the exercises below, the target signal is: [1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0]

Get a piece of graph paper (or print this) and prepare to plot by hand.

(1) Make a vertical axis one inch high that runs from 0 to 1 and a horizontal axis from 0 to 15. Suppose the estimate starts at 0 and the step-size (in the equation) is 0.5. (The estimate should be zero on the first time step.) Plot the trajectory of the estimate through t=15. How close is the estimate to 1.0 after the update at t=4? Suppose for a moment that the target signal continued to be 1.0. Without plotting, how close would it be at t = 10? At t=20?

(2) Start over with a new graph and the estimate again at zero. Plot again, this time with a step size of 1/8.

(3) Make a third graph with a step size of 1.0 and repeat. Which step size produces estimates of smaller error when the target is alternating (i.e., t=10 through t=15)?

(4) Make a fourth graph with a step size of 1/t (i.e., the first step size is 1, the second is 1/2, the third is 1/3, etc.) Based on these graphs, why is the 1/t step size appealing? Why is it not always the right choice?

(5) What happens if the step size is minus 0.5? 1.5? Plot the first few steps if you need to. What is the safe range for the step size?

The following part requires installation of OpenAI Gym(link), and the classical control domains(link).

In class, we compare two temporal difference learning based control: SARSA and Q. We will introduce another method called Expected SARSA. The difference of SARSA, Q, and Expected SARSA lies in the mechanism of choosing a_{t+1}.

SARSA:$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$, where $a_{t+1} \leftarrow \pi(a|s_{t+1})$ is chosen from s_t by using the current policy π.

Q: $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$ where the agent takes the maximum of the "next action" $a_{t+1} \leftarrow \arg\max_a Q(s_{t+1}, a)$.

Expected SARSA replaces the max operator with expectation operator:

Expected SARSA:$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \sum_a \pi(s_{t+1}, a) Q(s_{t+1}, a) - Q(s_t, a_t)]$ where the agent takes the expectation of the "next action".

3 Programming: Mountain car (15 points)

Mountain car is a benchmark domain in reinforcement learning, as can be seen here:

https://en.wikipedia.org/wiki/Mountain_Car

Run the experiments for 20 runs with Q-learning, SARSA, and expected SARSA (some algorithms are already included in Gym, which you can directly call and run, Hooray!), and compare their performance

(mean and variance), with the number of iterations (you can choose an appropriate number of iterations). You are responsible to tune the parameters to improve your performance, and give the performance analysis of different methods. **Your plot should show both mean curve and variance over 20 runs, with three curves:** Q-learning, SARSA, and expected SARSA**.**

4 (Bonus) Programming: CartPole(10 points)

Use REINFORCE algorithm to play with CartPole! Still 20 runs. **Your plot should show both mean curve and variance over 20 runs, with one curve: REINFORCE.**