CS5804 Homework 3

Homework must be submitted electronically following the instructions on the course homepage. Make sure to explain you reasoning or show your derivations. You will lose points for unjustified answers, even if they are correct.

Written Problems

1. (Based on R+N 17.4) Sometimes MDPs are formulated with a reward function $R(s, a)$ that depends on the action taken or with a reward function $R(s, a, s')$ that also depends on the outcome state.

   (a) (2 points) Write the Bellman equations for both of these formulations.

   (b) (3 points) Show how an MDP with reward function $R(s, a, s')$ can be transformed into a different MDP with reward function $R(s, a)$, such that optimal policies in the new MDP correspond exactly to optimal policies in the original MDP.

   (c) (3 points) Now do the same to convert MDPs with $R(s, a)$ into MDPs with $R(s)$.

2. Section 21.4 in R+N discusses more compact, approximate parameterizations for utility functions, using a linear combination of *basis feature functions* of the state:

$$\hat{U}^{\pi}_{\boldsymbol{\theta}}(s) := \sum_{i=1}^{D} \theta_i f_i(s).$$

   For this linear approximation, the TD-learning update for each parameter θ_i, when transitioning from s to s', is

$$\theta_i \leftarrow \theta_i + \alpha \left(R(s) + \gamma \hat{U}^{\pi}_{\boldsymbol{\theta}}(s') - \hat{U}^{\pi}_{\boldsymbol{\theta}}(s) \right) f_i(s)$$

   (a) (2 points) For any finite, discrete state space, describe a basis of feature functions for which this linear approximation is exactly TD-learning. Show how the approximate TD-learning update maps exactly to the full TD update

$$U^{\pi}(s) \leftarrow U^{\pi}(s) + \alpha(R(s) + \gamma U^{\pi}(s') - U^{\pi}(s))$$

   (b) (2 points) If the discrete state space is the 5×5 grid world below, describe how you can use a much more compact set of feature functions to represent states, and discuss how this compact form enables generalization to states your TD-learning agent has never visited.

-1	-1	-1	-1	-1
-1				-1
-1		+10		-1
-1				-1
-1	-1	-1	-1	-1

(Problem set continues on next page.)

3. (3 points. This question is an open-ended one, so you will get credit for thoughtful answers with analysis and justification.) Describe in a few paragraphs how you would set up an active reinforcement learning environment for a self-driving car.[1] What features would you include in the state-action representation? What actions would you allow the agent choose among? What would the reward function be? What do you think would work well in your setup, and what will not work as well? What are compromises you make in your design for computational efficiency?

   Be creative and have fun with this one. But make sure your answer demonstrates some technical depth in your ideas.

[1]Or if your own research investigates a different real-world application that reinforcement learning could work for, feel free to write about that application.