SYS 6582: Reinforcement Learning **Spring 2018**

Homework 3

Handed Out: March 12, 2018 *Due: March 26,* 2018

1. (Exercise 5.1, S&B) Consider the diagrams on the right in Figure 5.1. Why does the estimated value function jump up for the last two rows in the rear? Why does it drop off for the whole last row on the left? Why are the frontmost values higher in the upper diagrams than in the lower? (5%)

2. (Exercise 6.3, S&B) From the results shown in the left graph of the random walk example (on the next page) it appears that the first episode results in a change in only V(A). What does this tell you about what happened on the first episode? Why was only the estimate for this one state changed? By exactly how much was it changed? (10%)

3. (Exercise 6.6, S&B) In Example 6.2 we stated that the true values for the random walk example are $1/6, 2/6, 3/6, 4/6$, and $5/6$, for states A through E. Describe at least two different ways that these could have been computed. Which would you guess we actually used? Why? (10%)

4. (Exercise 7.3) Why do you think a larger random walk task (19 states instead of 5) was used in the examples of this chapter? Would a smaller walk have shifted the advantage to a different value of n? How about the change in left-side outcome from 0 to 1 made in the larger walk? Do you think that made any difference in the best value of n? (10%)

5. Implement the Monte Carlo every-visit policy evaluation algorithm (with incremental updating). Complete *MC_eVisit.py.* Then run *blackjack_prediction.py* with num_episode = 10,000 and 500,000 to test your code on the blackjack example. You should have figures similar to Figure 5.1 in the textbook. Submit the two figures and *MC_eVisit.py.* (15%)

6. Implement the TD(0) algorithm. Complete function *TD0* in *TD.py.* Then run *rw.py* to test your code on the random walk example (Example 6.2 in the textbook). Submit the two figures generated from this run. (10%)

7. Implement the n-step TD algorithm. Complete function *TD_n* in *TD.py.* Run *rw_nstepTD.py* to test your code on the large random walk example, i.e., Example 7.1 in the textbook. You should have a figure similar to (part of) Figure 7.2 in the textbook. Submit this figure on collab. (25%)

8. Implement the TD(λ) (backward view) algorithm. Complete function *TD_lambda* in *TD.py.* Run *rw_TDlambda.py* to test your code on the large random walk example. Your figure should be similar to the left one in Figure 12.6 in the textbook. Submit this figure, and the complete *TD.py* on collab. (15%)