NAME: Shaurya Seth

Written assignment w1 – Qualifying assignment

CMPUT 609:
Reinforcement Learning II

This assignment is meant to assess your preparation for taking the RLII class, that is, to assess your knowledge of Part 1 of the RL textbook. This assignment will be marked, but the assessment is primarily for your benefit. The RLII course will be much more useful to you if you have the right background. If you don't, then why not get it from the RLI course next fall and then take RLII next year? If all the students in the course have about the same level of preparation, then we will be able to teach and learn that much more efficiently.

If you have appropriate knowledge, you should be able to complete this assignment easily in 90 minutes without consulting the textbook. Try taking it as if it was a closed-book exam to see how well you can do. To the extent that you need to consult the textbook you should probably reconsider taking the course this year.

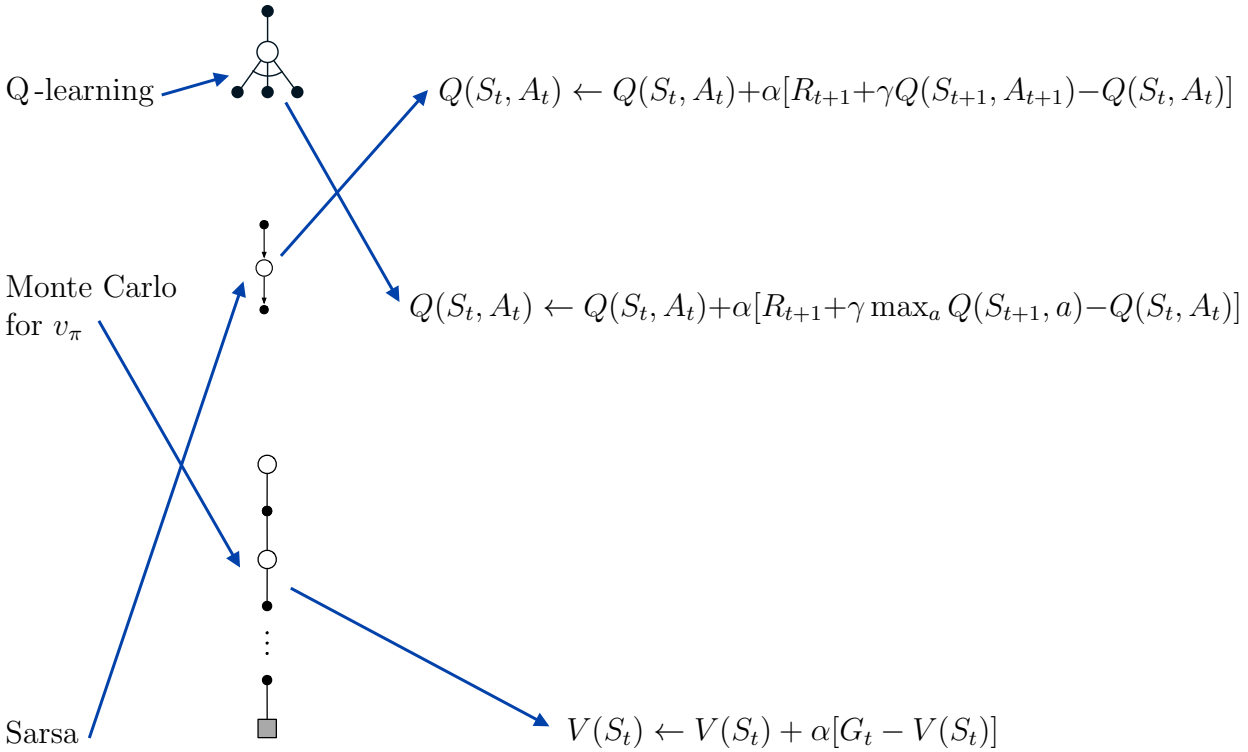
Start by writing down your name. Answer the questions directly on the assignment pages in the space provided. Attach additional pages if necessary. Show your work and how you arrive at your answers. You will be marked on both the final answer and how you obtained it, so go ahead and show your thought processes.

You will have 24 hours to complete the assignment. You can either use a pdf annotation tool to add your answers or, if you must, you can write by hand and then take photos of your pages, combine them into one file, and submit that.

Good luck!

1. (3 points) The primary goal of reinforcement learning can be seen as producing a policy, which maps from states to actions.

2. (6 pts total) Draw lines connecting the corresponding algorithm names, update diagrams, and update rules:



3. (3 pts) **Multiple choice:** In TD methods, a larger discount parameter γ, $0 < \gamma < 1$, means

   a) a closer approximation to the dynamic-programming solution

   b) less concern for immediate rewards relative to later rewards

   c) more concern for immediate rewards relative to later rewards

   d) both a) and b)

   e) both a) and c)

4. (6 pts total) This question has three parts, each of which should be answered concisely. In addition, provide a brief explanation or justification for your concise answer.

   Suppose you have a policy π and its action-value function, q_π, then you greedify q_π to produce a deterministic policy π' such that:

$$\pi'(s) = \arg\max_a q_\pi(s, a) \qquad \text{for all } s \in \mathcal{S}.$$

   (a) What do you know about the relationship between π and π'?

   pi' >= pi (because pi' will always take the best action under q_pi and obtain a greater or equal expected return)

   (b) Now suppose you notice that π' is the same as π. What then do you know about the two policies?

   pi and pi' are both optimal policies (because making the policy greedy with respect to q_pi is not improving it any further and so q_pi must be q*)

   (c) Now suppose you notice that π' is different from π. Do you know anything more about the two policies other than what you reported in part (a)?

   pi' > pi (this means pi was definitely not the optimal policy and q_pi was not q*)

5. (5 pts) Suppose the discount rate γ is 0.5 and the following sequence of rewards is observed: $R_1 = 2$, $R_2 = -8$, $R_3 = 4$, $R_4 = 16$, followed by the terminal state. What are the following returns?

   $G_4 =$ 0

   $G_3 =$ 16

   $G_2 =$ 12

   $G_1 =$ -2

   $G_0 =$ 1

6. (4 pts total) In *batch updating*, a fixed training set of data (experience interacting with an MDP) is presented over and over again to a learning algorithm, with updates made only after processing the whole set. For the tabular case and a sufficiently small fixed step size α, each algorithm will converge to a characteristic solution independent of α.

   True or False: Comparing tabular TD and Monte Carlo solutions under batch updating according to their empirical Mean Square Error (MSE):

   (a) **T** **F** (2 pts) MC solutions will always have the lowest MSE on the training data

   (b) **T** **F** (2 pts) TD solutions will usually have higher MSE on new data

7. (3 pts) In reinforcement learning we consider both learning methods and planning methods for solving MDPs. What is the essential difference between learning methods and planning methods?

   planning methods require a model of the environment (the agent uses the model to plan ahead before taking action)

8. All about q_* (20 points total)

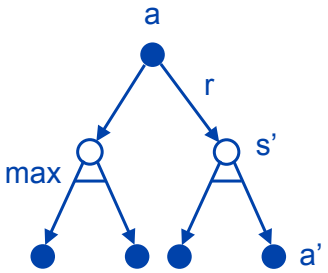
Consider the value function q_* for a continuing Markov decision process with discounting.

(a) [4 points] Give an equation *defining* $q_*(s, a)$ in terms of the subsequent rewards $R_{t+1}, R_{t+2}, \ldots$ given that you start, at time t, in state s taking action a. If you define q_* in terms of other things (e.g., q_π, G_t, or π_*) then be sure to define them in terms of the underlying rewards as well.

q*(s, a) = E[Rt+1 + γRt+2 + γ^2Rt+3 + … | St=s, At=a]

where E is expectation

(b) [3 points] Sketch the update diagram (also sometimes known as a backup diagram) for the dynamic programming algorithm for q_*. Find a place to attach the labels a, r, s', and a'.



(c) [5 points] What is the Bellman equation for q_*? Write it in an explicit form in terms of $p(s', r|s, a)$ so that no expected-value notation appears.

q*(s, a) = ∑ p(s', r |s, a) [r + γ max(a') q*(s', a')] for all possible s', r

γ is the discount factor

(d) [4 points] Consider the simplest dynamic-programming algorithm for computing q_*. An array $Q(s, a)$ is initialized to zero. Then there are repeated sweeps through the state–action space, with one update done for each state–action pair. What is that update?

$Q(S, A) = \sum p(s', r \mid s, a) [r + \gamma Q(s', A')]$ for all possible s', r

γ is the discount factor

(e) [4 points] Now consider a simple tabular temporal-difference learning method for estimating q_* from experience from an ε-greedy policy with $\varepsilon = 0.1$. An array $Q(s, a)$ is initialized to zero. Then an infinite sequence of experience, $S_0, A_0, R_1, S_1, A_1, R_2, S_2, \ldots$, is processed, with one update to $Q(S_t, A_t)$ done for each transition from S_t to S_{t+1}. What is that update?

$Q(S, A) = Q(S, A) + \alpha[R + \gamma Q(S', A') - Q(S, A)]$

A' is chosen using ε-greedy action selection
α is the step-size
γ is the discount factor

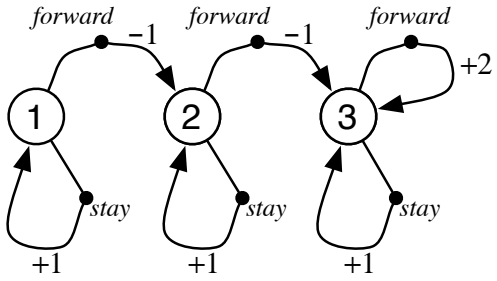
9. (5 pts) What are the key differences between on-policy and off-policy learning? (point form ok)

on-policy learning:
- agent learns from the current "target" policy it is following
- updates occur for the most recent trajectory

off-policy learning:
- agent can learn from past experience of another "behaviour" policy
- can use experience-replay buffers

10. Markov Decision Process (16 points total)

Consider the **continuing** finite MDP in the figure below. There are three states, (1, 2, and 3), and two actions, *forward* and *stay*. The *forward* action takes the agent to a higher numbered state (except in state 3), and the *stay* action keeps the agent in the same state. The effect of all actions is deterministic. The expected rewards on each transition are as indicated in the figure.



using the bellman optimality equation starting from state 3 and moving backwards

(a) (6 points) Suppose the discount factor is $\gamma = \frac{1}{3}$. What then is the optimal value function and the optimal deterministic policy?

$v_*(1) =$ 5/3 $\quad \pi_*(1) =$ forward
$v_*(2) =$ 2 $\quad \pi_*(2) =$ forward
$v_*(3) =$ 3 $\quad \pi_*(3) =$ forward

(b) (6 points) Suppose the discount factor is $\gamma = \frac{4}{5}$. What then is the optimal value function and the optimal deterministic policy?

$v_*(1) =$ 41/5 $\quad \pi_*(1) =$ forward
$v_*(2) =$ 9 $\quad \pi_*(2) =$ forward
$v_*(3) =$ 10 $\quad \pi_*(3) =$ forward

(c) (4 points) Suppose you are doing learning by the tabular TD(0) algorithm. You start with all value estimates $V(s) = 0$, and you observe the following partial trajectory (sequence of states, actions and rewards, where the state numbers are bolded):

1, *forward*, -1, **2**, *stay*, +1, **2**

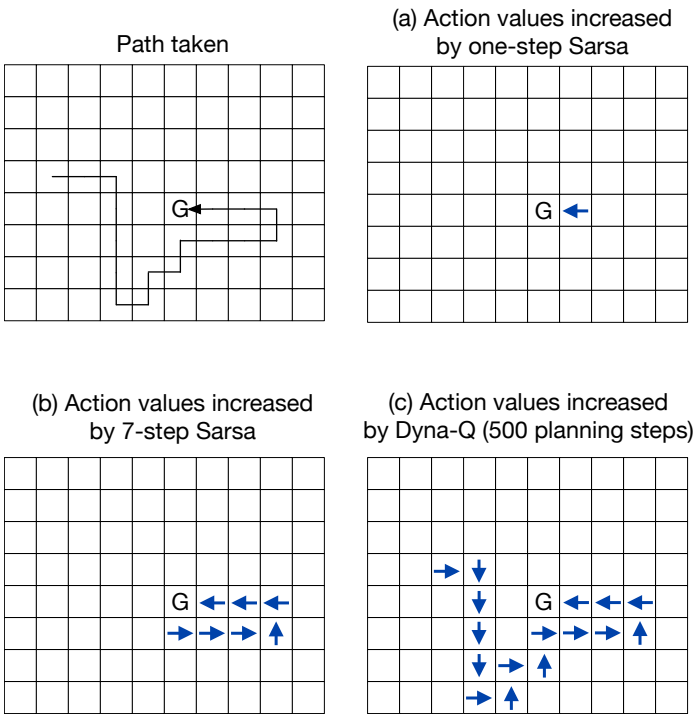
Assuming the step size is $\alpha = 0.25$, and $\gamma = 0.5$, show the updates that are performed.

V(1) = -0.25
V(2) = 0.25

11. Multi-step updates (6 points)

Consider the gridworld depicted in the diagram below. There are four actions corresponding to up, down, right, and left movements. The first panel shows the path taken by an agent in a single episode, ending at a location of high reward, marked by the G. In this example the values were all zero at the start of the episode, and all rewards were zero during the episode except for a positive reward at G.

Your job is to to complete the diagrams in panels (a), (b), and (c). Draw arrows in the correct grid squares to show which action values were increased by the end of the episode by (a) one-step Sarsa, (b) 7-step Sarsa, and (c) the Dyna-Q method with 500 steps of planning. In order to illustrate that an action-value was increased make an arrow in the state, with the direction corresponding to that action. For example, if `up` action was increased in a state we would mark that as:  , and if `right` action was increased in a state we would mark that as:  .



For part (c) please explain your answer:

the agent has only seen one trajectory and can only make action value updates in the states it has visited. planning can help these updates but need in finding a better path

12. (5 pts) For a finite continuing discounted MDP with discount factor γ, suppose you know two numbers $r_{\min}$ and $r_{\max}$ such that for all immediate rewards r, $r_{\min} \leq r \leq r_{\max}$. Give epresssions for two numbers $V_{\min}$ and $V_{\max}$ such that $V_{\min} \leq V^{\pi}(s) \leq V_{\max}$ for all states $s \in S$ and all policies π.

V_min corresponds to receiving r_min at every timestep
V_min = r_min / (1-γ)

V_max corresponds to receiving r_max at every timestep
V_max = r_max / (1-γ)

13. (2 pts) **True or False and justify your answer**:

If a policy is greedy with respect to the value function for the equiprobable random policy, then it is an optimal policy.

False (the greedy policy may be an improvement over the random policy but is not guaranteed to be optimal as random action selection does not give an optimal value function)