CS 4180/5180: Reinforcement Learning and Sequential Decision Making (Fall 2024) — Lawson L.S. Wong
Northeastern University — Due February 2, 2024

Exercise 2: Markov Decision Processes

Please remember the following policies:

- Exercises are due at **11:59 PM** Boston time (ET).
- **Submissions should be made electronically on Canvas.**
  Please ensure that your solutions for both the written and programming parts are present.
  Upload all files in a single submission, keeping the files individual (do not zip them).
  You can make as many submissions as you wish, but only the latest one will be considered, and late days will be computed based on the latest submission.
- Each exercise may be handed in up to two days late (24-hour period), penalized by 5% per day.
  Submissions later than this will not be accepted. There is no limit on the total number of late days used over the course of the semester.
- Written solutions may be handwritten or typeset. For the former, please ensure handwriting is legible. If you write your answers on paper and submit images of them, that is fine, but please put and order them correctly in a single .pdf file. One way to do this is putting them in a Word document and saving as a PDF file.
- Programming solutions should be in Python 3, either as `.py` or `.ipynb` files.
  For the latter (notebook format), please export the output as a PDF file and submit that together. To do so, first export the notebook as an HTML file (File: Save and Export Notebook as: HTML), then open the HTML file in a brower, and finally print it as a PDF file.
- You are welcome to discuss these problems with other students in the class, but you must understand and write up the solution **and code** yourself, *and* indicate who you discussed with (if any). If you are collaborating with a large language model (LLM), acknowledge this and include your entire interaction with this system. We strongly encourage you to formulate your own answers first before consulting other students / LLMs.
- Typically, each assignment contains questions designated [CS 5180 only.], which are required for CS 5180 students. Students in CS 4180 can complete these questions for extra credit, for the same point values. Occasionally, there will also be [Extra credit.] questions, which can be completed by everyone. Point values for these questions depend on the depth of the extra-credit investigation.
- Contact the teaching staff if there are *extenuating* circumstances.

1. **1 point.** *Formulating an MDP.*
   Written: It is instructive to formally define an MDP at least once, in particular, the dynamics function. Consider the four-rooms domain from Ex0.
   (a) What are the state and action spaces S, A?
   (b) Consider the dynamics function $p(s', r|s, a)$. Approximately how many non-zero rows are in this conditional probability table?
   (c) **Written or code:** Provide pseudocode or actual code to generate the full table of non-zero $p(s', r|s, a)$ values. Notes:
   - If you provide pseudocode, it should be more specific than "generate $p(s', r|s, a)$ for all (s, a, s', r)". It should include some actual probability values, consider effects of walls, etc.
   - If you generate the table using code, please include both the code and the generated table of values.
   - Recall: The goal state is at $(10, 10)$. Any action from there teleports the agent back to $(0, 0)$.

2. **1 point.** (RL2e 3.6, 3.7) *The RL objective.*
Written:

   (a) Suppose you treated pole-balancing as an episodic task but also used discounting, with all rewards zero except for -1 upon failure. What then would the return be at each time? How does this return differ from that in the discounted, continuing formulation of this task? (Derive expressions for both the episodic and continuing cases.)

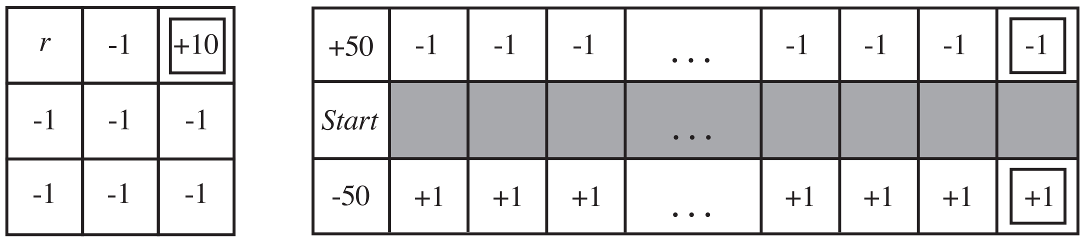
   (b) Imagine that you are designing a robot to run a maze. You decide to give it a reward of $+1$ for escaping from the maze and a reward of zero at all other times. The task seems to break down naturally into episodes – the successive runs through the maze – so you decide to treat it as an episodic task, where the goal is to maximize expected total reward (Equation 3.7). After running the learning agent for a while, you find that it is showing no improvement in escaping from the maze. What is going wrong? Have you effectively communicated to the agent what you want it to achieve?

3. **1 point.** (RL2e 3.8, 3.9) *Discounted return.*
Written:

   (a) Suppose $\gamma = 0.5$ and the following sequence of rewards is received:
   $R_1 = -1, R_2 = 2, R_3 = 6, R_4 = 3, R_5 = 2$
   with $T = 5$. What are $G_0, G_1, \ldots, G_5$? Hint: Work backwards.

   (b) Suppose $\gamma = 0.9$ and the reward sequence is $R_1 = 2$ followed by an infinite sequence of '7's.
   What are G_1 and G_0?

4. **1 point.** (Artificial Intelligence: A Modern Approach 17.9.) *Discount factor.*



(a) (b)

Written: Consider the 101×3 world shown above (right; the left sub-figure will not be used). In the start state the agent has a choice of two deterministic actions, `Up` or `Down`, but in the other states the agent has one deterministic action, `Right`. Express the value of each action as a function of the discount factor γ. For what values of the discount factor γ should the agent choose `Up` and for which `Down`? It is fine to leave your answer as an unsolved expression, although you can solve for the threshold value of γ numerically using tools such as WolframAlpha.
(This simple example reflects real-world situations in which one must weigh the value of an immediate action versus the potential continual long-term consequences, such as choosing to dump pollutants into a lake.)

5. **1 point.** (RL2e 3.15, 3.16) *Modifying the reward function.*
Written:

   (a) In the gridworld example, rewards are positive for goals, negative for running into the edge of the world, and zero the rest of the time. Are the signs of these rewards important, or only the intervals between them? Prove, using Equation 3.8, that adding a constant c to all the rewards adds a constant, v_c, to the values of all states, and thus does not affect the relative values of any states under any policies. What is v_c in terms of c and γ?

   (b) Now consider adding a constant c to all the rewards in an episodic task, such as maze running. Would this have any effect, or would it leave the task unchanged as in the continuing task above? Why or why not? Give an example.

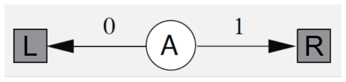
6. **1 point.** (RL2e 3.14) *Bellman equation.*
Written:

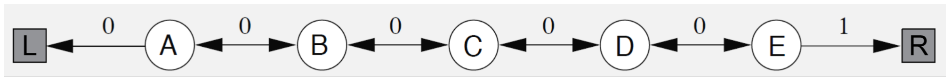
(a) The Bellman equation (Equation 3.14) must hold for each state for the value function v_π shown in Figure 3.2 (right) of Example 3.5. Show numerically that this equation holds for the center state, valued at $+0.7$, with respect to its four neighboring states, valued at $+2.3, +0.4, -0.4, +0.7$. (These numbers are accurate only to one decimal place.) Note that Figure 3.2 (right) is the value function for the equiprobable random policy.

(b) The Bellman equation holds for *all* policies, including optimal policies. Consider v_* and π_* shown in Figure 3.5 (middle, right respectively). Similar to the previous part, show numerically that the Bellman equation holds for the center state, valued at $+17.8$, with respect to its four neighboring states, for the optimal policy π_* shown in Figure 3.5 (right).

If the Bellman equation is unclear to you, you should practice this question again for other states.

7. **1 point.** *Guessing and verifying value functions.*





Written: We have not discussed algorithms for finding the value function yet, but it is possible to guess them for simple problems and verify that they are consistent with the Bellman equation. A purported value function $V(s)$ is consistent (according to the Bellman equation) for all states under the policy π *if and only if* $V = v_\pi$, the unique value function for π.

(a) Consider the simple 3-state MDP shown above (left). All episodes start in the center state, A, then proceed either left or right by one state on each step, with equal probability. Episodes terminate either on the extreme left (L) or the extreme right (R). When an episode terminates on the right, a reward of $+1$ occurs; all other rewards are zero. This is an undiscounted MDP ($\gamma = 1$). Guess the value function for this MDP (for the equiprobable random policy), and verify its consistency using the Bellman equation.

(b) Now consider an extension of the MDP to contain 7 states total (right). Guess the value function for this MDP (for the equiprobable random policy), and verify its consistency using the Bellman equation.
Hint: The new value function has a simple form. Use part (a) to inform your thinking.

(c) What do you think the value function is for arbitrary an arbitrary number of states n?

8. **[CS 5180 only.]** **2 points.** *Solving for the value function.*
Written: Another way to solve for the value function is to write out the Bellman equation for each state, view it as a system of linear equations, and then solve for the unknowns (the value of each state). Consider a particularly simple MDP, the 2-state recycling robot in Example 3.3.

(a) Expand the Bellman equation for the 2 states in the recycling robot, for an arbitrary policy $\pi(a|s)$, discount factor γ, and domain parameters $\alpha, \beta, r_{\texttt{search}}, r_{\texttt{wait}}$ as described in the example.

(b) You should now have two linear equations involving two unknowns, $v(\texttt{high})$ and $v(\texttt{low})$, as well as involving the policy $\pi(a|s)$, γ, and the domain parameters. Let $\alpha = 0.8, \beta = 0.6, \gamma = 0.9, r_{\texttt{search}} = 10, r_{\texttt{wait}} = 3$.
Consider the policy $\pi(\texttt{search} \mid \texttt{high}) = 1$, $\pi(\texttt{wait} \mid \texttt{low}) = 0.5$, and $\pi(\texttt{recharge} \mid \texttt{low}) = 0.5$.
Find the value function for this policy, i.e., solve the equations for the values of $v(\texttt{high})$ and $v(\texttt{low})$.
Check that your solution satisfies the Bellman equation.

(c) **[Extra credit.]** **1 point.** Suppose you can modify the policy in the low state, i.e., set $\pi(\texttt{wait} \mid \texttt{low}) = \theta$, and $\pi(\texttt{recharge} \mid \texttt{low}) = 1 - \theta$. What θ should you set it to, and what is the value function for that θ?