
CMPUT 261, Winter 2023
Assignment #4

Due: Tuesday, April 11/2023
Total points: 101

For this assignment use the following consultation model:

1. you can discuss assignment questions and exchange ideas with other *current* CMPUT 261 students;
2. you must list all members of the discussion in your solution;
3. you may **not** share/exchange/discuss written material and/or code;
4. you must write up your solutions individually;
5. you must fully understand and be able to explain your solution in any amount of detail as requested by the instructor and/or the TAs.

Anything that you use in your work and that is not your own creation must be properly cited by listing the original source. Failing to cite others' work is plagiarism and will be dealt with as an academic offence.

First name: Dipesh

Last name: Patel

CCID: dipesh1 @ualberta.ca

---

1. (**Markov Decision Processes**; 15 points)

   Two coins are placed in a tray; each coin is either heads up or tails up with equal probability, independent of the other coin. A robot arm starts above **one** of the coins, again with equal probability. At each time step, the arm can perform one of the following operations:

   - Flip the coin that it is above: The coin under the arm will be randomly set to either heads up or tails up with equal probability, and a reward of -1 is generated.
   - Move to be above the other coin; this generates a reward of -2.
   - Call for evaluation; a reward of 15 is generated if both coins are heads up; a reward of 10 is generated if both coins are tails up; or a reward of -8 if the coins mismatch. After the reward is generated, both coins are flipped.

   (a) **[10 points]** Represent this scenario formally as a Markov Decision Process.

{ Robot arm location, side up, side up }

States: { (right, head, head), (left, head, head),
(right, head, tail), (left, head, tail),
(right, tail, head), (left, tail, head),
(right, tail, tail), (left, tail, tail) }

Actions: { Flip, move, call }

Rewards: { Flip: −1, move: −2, call: if both tails up: 10; if both heads up: 15; if different sides up: −8 }

Dynamics:

S	Action(a)	S'	P(s'|s,a)	R(s,a,s')
(right, head, head)	Flip	(right, head, tail)	0.5	−1
(right, head, tail)	Flip	(right, head, tail)	0.5	−1
(right, tail, head)	Flip	(right, tail, head)	0.5	−1
(right, tail, tail)	Flip	(right, tail, tail)	0.5	−1
(left, head, head)	Flip	(left, tail, head)	0.5	−1
(left, head, tail)	Flip	(left, head, tail)	0.5	−1

(left, tail, head)	Flip	(left, tail, head	0.5	-1
(left, tail, tail)	Flip	(left, head, tail)	0.5	-1
⋮	⋮	⋮	⋮	⋮
(right, head, head)	move	(left, head, head)	1	-2
(right, head, tail)	move	(left, head, tail)	1	-2
(right, tail, head)	move	(left, tail, head)	1	-2
(right, tail, tail)	move	(left, tail, tail)	1	-2
(left, head, head)	move	(right, head, head)	1	-2
(left, head, tail)	move	(right, head, tail)	1	-2
(left, tail, head)	move	(right, tail, head)	1	-2
(left, tail, tail)	move	(right, tail, tail)	1	-2

(right, head, head)	call	(right, tail, tail)	1	15
(right, head, tail)	call	(right, tail, head)	1	-8
(right, tail, head)	call	(right, head, tail)	1	-8
(right, tail, tail)	call	(right, head, head)	1	10
(left, head, head)	call	(left, tail, tail)	1	15
(left, head, tail)	call	(left, tail, head)	1	-8
(left, tail, head)	call	(left, head, tail)	1	-8
(left, tail, tail)	call	(left, head, head)	1	10

(b) **[2 points]** Is this a continuing or episodic scenario? Justify your answer.

This is a continuing scenario because scenario never ends. Even after "call" the coins are flipped and scenario continues. In other words scenario doesn't have terminal state.

(c) **[3 points]** If you answered episodic to the previous question, describe how to treat the scenario as continuing. If you answered continuing to the previous question, describe how to treat the scenario as episodic.

To have episodic scenario we need a terminal state.

we can have any two integers, let's say 16 & 100, and set the scenario such that if you have rewards less than 16 or more than 100 the scenario ends.

2. (**Bellman Action-Value Equation**; 8 points)

   (a) **[4 points]** Give an expression for $q_\pi(s,a)$ as an expectation of random variables, in the same form as equation (4.3) in the text Sutton & Barto, *Reinforcement Learning: An Introduction, 2nd edition.*

$$q_\pi(s,a) = E_\pi\left[R_{t+1} + \gamma \sum_{A_{t+1}} \pi(A_{t+1} \mid S_{t+1})\, q_\pi(S_{t+1}, A_{t+1}) \;\middle|\; S_t = s, A_t = a \right]$$

   (b) **[4 points]** Give an expression for $q_\pi(s,a)$ as a weighted sum, in the same form as equation (4.4) in the text Sutton & Barto, *Reinforcement Learning: An Introduction, 2nd edition.*

$$q_\pi(s,a) = \sum_{s_{t+1}, r} p(s', r \mid s, a)\left[r + \gamma \sum_{a_{t+1}} \pi(a_{t+1} \mid s')\, q_\pi(s', a_{t+1}) \right]$$

3. (**Policy Improvement**; 12 points) Consider the following MDP with actions $\mathcal{A} = \{a, b\}$, states $\mathcal{S} = \{W, X, Y, Z\}$, and the following dynamics. All unspecified transitions have probability 0:

$$\begin{array}{lll} p(X,4|W,a) = 0.5 & p(Z,0|X,a) = 0.8 & p(Z,1|Y,a) = 1 \\ p(Y,2|W,a) = 0.5 & p(Z,25|X,a) = 0.2 & p(Z,5|Y,b) = 1 \\ p(Z,3|W,b) = 1 & p(Z,0|X,b) = 1 & p(Z,0|Z,a) = 1 \\ & & p(Z,0|Z,b) = 1 \end{array}$$

   (a) **[4 points]** Consider the policy $\pi(b|s) = .5, \pi(a|s) = .5$ for all states $s \in \mathcal{S}$; i.e., a policy that randomizes between the two actions uniformly. Using a discount rate of $\gamma = 0.8$, what is the value $v_\pi(s)$ for each state $s \in \mathcal{S}$? (Hint: What must the value of state Z be under any policy?)

$v_\pi(W) = 0.5 \times 0.5 \times (4 + 0.8 \cdot v_\pi(X)) + 0.5 \times 0.5 (2 + 0.8 \cdot v_\pi(Y)) + 0.5 (3 + 0.8 \cdot v_\pi(Z)) = 4.1$

$v_\pi(X) = 0.5 \times 0.8 \times (0 + 0.8 \cdot v_\pi(Z)) + 0.5 \times 0.2 (25 + 0.8 \cdot v_\pi(Z)) + 0.5 \times 1 (0 + 0.8 \cdot v_\pi(Z)) = 2.5$

$v_\pi(Y) = 0.5 \times 1 (1 + 0.8 \cdot v_\pi(Z)) + 0.5 \times 1 (5 + 0.8 \cdot v_\pi(Z)) = 3$

Let $v_\pi(Z) = 0.5 \times 1 (0 + 0.8 \cdot v_\pi(Z)) + 0.5 \times 1 (0 + 0.8 \cdot v_\pi(Z)) = 0$

Thus, having $v_\pi(Z) = 0$, we get $v_\pi(W) = 4.1$, $v_\pi(Y) = 3$, $v_\pi(X) = 2.5$, $v_\pi(Z) = 0$

   (b) **[8 points]** Construct a policy π' that strictly improves upon π: That is, $v_{\pi'}(s) \geq v_\pi(s)$ for all $s \in \mathcal{S}$, and $v_{\pi'}(s) > v_\pi(s)$ for at least one $s \in \mathcal{S}$. Appeal to the Policy Improvement Theorem to show that your new policy is indeed a strict improvement.

Let's suppose π': $\pi'(a|w) = 1$, $\pi'(a|x) = 1$, $\pi'(b|y)$, $\pi'(a|z) = 1$

Then, $v_{\pi'}(w) = 1 \times 0.5 (4 + 0.8 \cdot v_{\pi'}(x)) + 1 \times 0.5 (2 + 0.8 \cdot v_{\pi'}(y)) = 7$

$v_{\pi'}(x) = 1 \times 0.8 (0 + 0.8 \cdot v_{\pi'}(z)) + 1 \times 0.2 (25 + 0.8 \cdot v_{\pi'}(z)) = 5$

$v_{\pi'}(y) = 1 \times 1 (5 + 0.8 \cdot v_{\pi'}(z)) = 5$

$v_{\pi'}(z) = 1 \times 1 (0 + 0.8 \cdot v_{\pi'}(z)) = 0$

Let $v_{\pi'}(z) = 0$

$7 > 4.1$, $5 > 2.5$, $5 > 3$, $0 \geq 0$ $\Rightarrow$ $v_{\pi'}(w) > v_\pi(w)$, $v_{\pi'}(x) > v_\pi(x)$, $v_{\pi'}(y) > v_\pi(y)$, $v_{\pi'}(z) \geq v_\pi(z)$

Thus, with policy improvement theorem, new policy is indeed strict improvement.

4. (**Monte Carlo Prediction**; 8 points) Consider an MDP with actions $\mathcal{A} = \{a, b, c\}$, states $\mathcal{S} = \{W, X, Y, Z\}$ (with terminal state Z), and unknown dynamics. Suppose that you have used a policy π to generate 4 episodes with the following trajectories:

$$\langle S_0 = W, A_0 = a, R_1 = 0, S_1 = X, A_1 = a, R_2 = 10, S_2 = Y, A_2 = b, R_3 = 0, S_3 = Z\rangle,$$
$$\langle S_0 = W, A_0 = a, R_1 = -10, S_1 = X, A_1 = b, R_2 = 0, S_2 = Z\rangle,$$
$$\langle S_0 = W, A_0 = b, R_1 = 2, S_1 = Y, A_1 = c, R_2 = 6, S_2 = Z\rangle,$$
$$\langle S_0 = W, A_0 = b, R_1 = 0, S_1 = Y, A_1 = c, R_2 = 12, S_2 = Z\rangle.$$

(a) **[8 points]** Use first-visit Monte Carlo prediction to estimate $v_\pi(s)$ for every $s \in \mathcal{S}$. Assume undiscounted rewards (i.e., $\gamma = 1$). $S = \{W, X, Y, Z\}$

$v_\pi(W) = \frac{10 + (-10) + 8 + 12}{4} = 5$

$v_\pi(X) = \frac{10 + 0}{2} = 5$

$v_\pi(Y) = \frac{0 + 6 + 12}{3} = 6$

$v_\pi(Z) = 0$

	W	X	Y	Z
epi:1	10	10	0	0
epi:2	-10	0		0
epi:3	8		6	0
epi:4	12		12	0

5. (**Temporal Difference Control**; 12 points) Consider an MDP with actions $\mathcal{A} = \{a, b, c\}$, states $\mathcal{S} = \{W, X, Y\}$, and unknown dynamics.

Suppose that you have previously computed the following estimated action values:

$Q(W, a) = 0$	$Q(X, a) = 8$	$Q(Y, a) = 0$
$Q(W, b) = 0$	$Q(X, b) = 7$	$Q(Y, b) = 0$
$Q(W, c) = 0$	$Q(X, c) = 16$	$Q(Y, c) = 0$

Now suppose that starting from state $S_t = W$, the current behaviour policy selects action $A_t = a$, leading to reward $R_{t+1} = 4$ and a transition to state $S_{t+1} = X$. The behaviour policy then selects action $A_{t+1} = a$.

(a) **[6 points]** What is the updated estimate for $Q(W, a)$ according to the Q-learning algorithm? Assume a step size of $\alpha = 0.5$ and a discount rate of $\gamma = 1$.

$Q(w,a) \leftarrow Q(w,a) + \alpha\left(R + \gamma \cdot \max_a Q(x,a) - Q(w,a)\right)$

$\leftarrow 0 + 0.5(4 + 1 \cdot 16 - 0)$

$\leftarrow 0.5(20)$

$\leftarrow 10$

Thus, updated estimate for $Q(w,a)$ according to Q-learning algorithm is 10

(b) **[6 points]** What is the updated estimate for $Q(W, a)$ according to the Sarsa algorithm? Assume a step size of $\alpha = 0.5$ and a discount rate of $\gamma = 1$.

$Q(w,a) \leftarrow Q(w,a) + \alpha\left(R + \gamma Q(w,a) - Q(w,a)\right)$

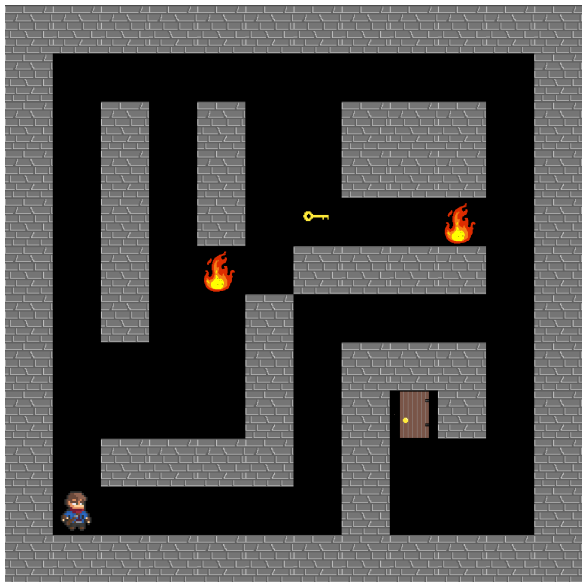
$\leftarrow 0 + 0.5(4 + 1 \cdot 8 - 0)$

$\leftarrow 0.5(12)$

$\leftarrow 6$

Thus, updated estimate for $Q(w,a)$ according to sarsa algorithm is 6

6. (**Implementation**; 46 points) Consider the following task (pictured below): an intrepid adventurer must make his way through a maze. To do so, he must first collect a key, located somewhere in the maze, before leaving through a door. Without the key, the door will not open. Additionally, the maze contains dangerous fire hazards, which the adventurer wishes to avoid.



This environment can be modelled as an MDP: the states correspond to the adventurer's (x, y) position in the maze and whether he currently possesses the key. Rewards are 0 on every time step the agent has not reached the end with the key, 1 for successfully completing the maze, and -100 if the agent touches fire. The agent would like to finish the maze as quickly as possible, so the environment's discount rate is $\gamma = 0.95$.

This question requires several external Python libraries. You can install them using the command

```
pip3 install --user -r requirements.txt
```

from the assignment directory.

(a) [**40 points**] Implement Q-learning by completing the implementations of the functions in the `agent.py` source file. An implementation of the environment, main training loop, and an action-value table are provided. The agent should use an epsilon-greedy behavior policy to ensure adequate exploration of the maze. The constants γ, ϵ and step size are already given as attributes of the agent, please use these in your implementation.

We have provided public test cases for this assignment; run them using `python3 tests.py`. (These tests take approximately 4 minutes to run on my laptop.)

We will test your program by running (a copy of) the module defined in `main.py`. You can run the same program using `python3 main.py`. An automated graphic will display a final episode after training has been completed, and a learning curve, showing the length of episodes over time will be generated in `learning_curve.png`.

(b) **[3 points]** In the function `display_final_episode()` the agent is evaluated using a *greedy* policy, instead of an ϵ-greedy policy. Why does it make sense to evaluate the agent this way? Would this still work with Sarsa?

we can evaluate agent this way because, learned policy's true performance is determined by learned policy itself. Because we need Q-value near optimal value.

Sarsa would also work as in the end Q-values be near optimal.

(c) **[3 points]** Could Monte Carlo control be used instead to solve this problem? Justify your answer.

yes, monte carlo can be used to solve this problem as we have both starting and terminal state i.e the scenerio is an episodic scenerio.

Submission

The assignment you downloaded from eClass is a single ZIP archive which includes this document as a PDF *and* its LaTeX source as well as a Python file needed for Question 6. You are to unzip the archive into an empty directory, work on the problems and then zip the directory into a new single ZIP archive for submission.

Each assignment is to be submitted electronically via eClass by the due date. **Your submission must be a single ZIP file containing**:

1. a single PDF file with your answers;
2. file(s) with your Python code.

To generate the PDF file with your answers you can do any of the following:

- insert your answers into the provided LaTeX source file between `\begin{answer}` and `\end{answer}`. Then run the source through LaTeX to produce a PDF file;
- print out the provided PDF file and legibly write your answers in the blank spaces under each question. Make sure you write as legibly as possible for we cannot give you any points if we cannot read your hand-writing. Then scan the pages and include the scan in your ZIP submission to be uploaded on eClass;
- use your favourite text processor and type up your answers there. Make sure you number your answers in the same way as the questions are numbered in this assignment.