

# PSet 5 – CS 4649/7649

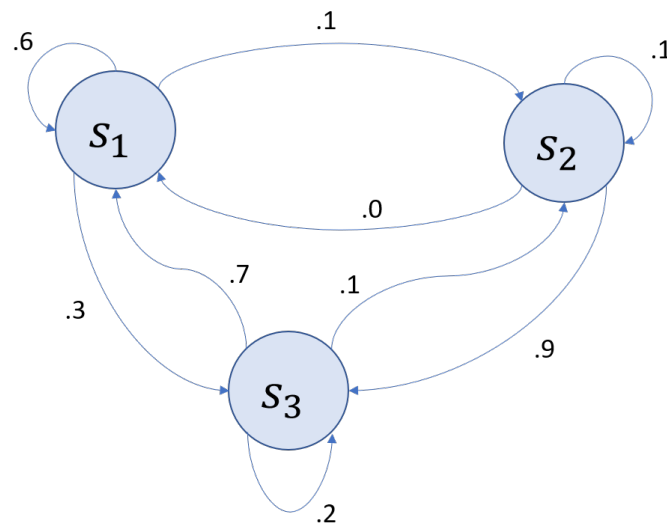
CS 4649/7649 Robot Intelligence: Planning  
Instructor: Matthew Gombolay

## Instructions:

- You may work with one or more classmates on this assignment. However, all work must be your own, original work (i.e., no copy+pasting code). You must list all people you worked with and sources you used on the document you submit for your homework
- Solutions to “by hand” problem must be enclosed by a box and be legible.
- Submit a PDF with Problems 1 and 2 to “Pset5 – written” and your Python code for Problems 3 and 4 to “Pset5 – coding” on Gradescope.

## Problem 1:

Consider the Markov Chain shown below. Compute the steady-state probability,  $\pi_i$ , of being in state  $i$  for each state  $i$ . You do not need to solve this by hand – you may use a graphic calculator or computer software. However, you must give an exact expression below that is “calculator ready.” For example, if you are going to plug into your calculator  $x = A^{-1}b$ , you must write out this equation as well as what  $A$  and  $b$  are; then, you must write out what the solution,  $x$ , is returned by your calculator. Place a box around the “calculator ready” equation as well as the solution. If multiple answers to  $x$  are possible, state which one is the one and only correct solution for this problem.



## Problem 2:

Write and solve a system of equations to compute the value function,  $V^\pi(s)$  for the MDP defined here:

- $S = \{s_1, s_2, s_3, s_4\}$

- $A = \{a_1, a_2\}$

- $T(s'|s, a_1) = \begin{bmatrix} .1 & .2 & .6 & .1 \\ .3 & .2 & .3 & .2 \\ .5 & .1 & .3 & .1 \\ .2 & .3 & .4 & .1 \end{bmatrix}$

- $T(s'|s, a_2) = \begin{bmatrix} .3 & .2 & .4 & .1 \\ .3 & .2 & .3 & .2 \\ .2 & .1 & .6 & .1 \\ .3 & .3 & .3 & .1 \end{bmatrix}$

- $\gamma = 0.95$

- $R(s, a) = \begin{bmatrix} -.1 & -.1 \\ -.1 & -.1 \\ -.1 & 1 \\ 1 & -.1 \end{bmatrix}$

- $\pi(s, a) = \begin{bmatrix} .9 & .1 \\ .2 & .8 \\ .3 & .7 \\ 0 & 1 \end{bmatrix}$

Note that the transition matrix,  $T(s'|s, a_1)$ , at element *row, col* is the probability of transitioning from state *row* to state *col*.

To receive full credit, write and place a box around your system of equations. Then, write and place a box around the values for  $V^\pi(s)$  for each state,  $s$ . You do not need to solve the equations by hand.





### Problem 3:

Using the same problem definition as Problem 2, ignoring the policy definition,  $\pi$ , implement in Python the Value Iteration procedure to find the optimal policy,  $\pi^*$ . Please use the Python template provided (pset5\_p3.py) and submit your code to Gradescope. We may test your code with other values as well.

### Problem 4 (7649 ONLY):

Implement Q-Learning in Python using the environment below and return the Q-table. Please use the Python template provided (pset5\_p4.py) and submit your code to Gradescope.

The map is a 4x4 grid. The states are the current location (row, col) (ex. Start is (0,0), goal is (2,2), lava is (2,1) and (1,2). The actions are up, right, left, down (0, 1, 2, 3 respectively). If you are on the edges of the map, and action might cause you to fall off the map. For example, the action up at (0,0). Falling off the map will get a reward of -2. The discount factor is .95. The map is static, but the actions are not-deterministic (with epsilon-greedy, epsilon=.2). We may also test your code using differently placed rewards. Set the learning rate  $\alpha = 1$ .

	0	1	2	3
0	Start 			
1			Lava: -1 	
2		Lava: -1 	Goal: +1 	
3				