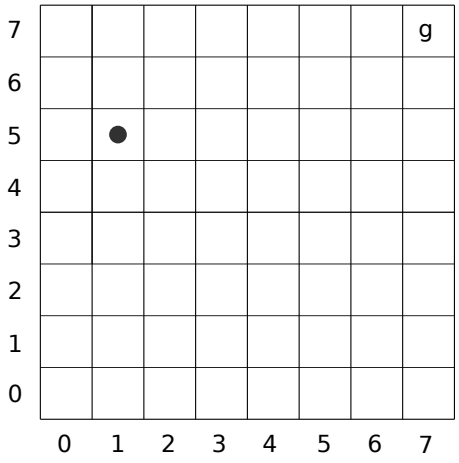
ME/ECE/MATH 497/597: Optimization Theory and Practice
Fall 2023 | Project I: Linear Programming

In this project, we will solve a few linear programs that finds the cost-to-go functions for a robot that lives in a gridworld.

1. [50 points] On an 8×8 chess board whose squares are coded from 0 to 7 along the horizontal and vertical directions, we have a robot, whose state is described the coordinate of the square it occupies.

   A remote-controller has 4 buttons that can move the robot `up`, `right`, `down`, and `left` by 1 square. At the start, we do not know which button maps to which of these directions. Furthermore, the microcontroller of the remote-controller has malfunctioned in such a way that every time one of the buttons is pressed, its mapping gets shuffled according to a uniform distribution. Under these conditions, we want to figure out what the expected value of the number of times the buttons need to be pressed in order to move the robot from any initial state to the goal state $g = (7, 7)$.

   Note: If the robot attempts to move out of bounds it will stay where it is.

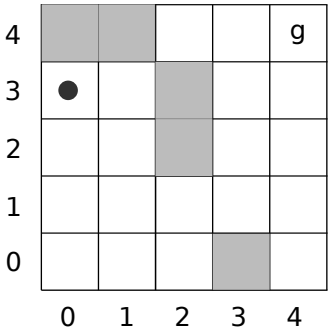
   (a) [20 points] Construct an LP whose solution yields the desired expected value.

   (b) [30 points] Solve the LP using your favorite programming language (Python recommended) to obtain this value. Interpret the results. Report the expected value for each state (square) as well as the sum of all expected values over all states.

   (c) [5 points (bonus)] Perform a Monte Carlo simulation that simulates the robot 500 times from each state until it reaches the goal position. Then compute the expected value by taking the average of all these runs. Compare the results with part (b).

2. [50 points] Obstacles are depicted by filled gray squares and their dynamics behave the same way as the bounds in question 1.

   Consider the same problem as in question 1, but now the board is 5×5 and some of the squares on the board are occupied such that the robot may not move into them as seen in the figure on the right.

   (a) [20 points] Repeat part (a) of question 1 for this setup.

   (b) [30 points] Repeat part (b) of question 1 for this setup.

   (c) [5 points (bonus)] Repeat part (c) of question 1 and compare the expected values you get with part (b).