Project #2

Amir Sadovnik
ECE 414/517: Reinforcement Learning (Fall 2021)

---

1 Overview

The goal of this project is to experiment by comparing Q-learning and ϵ-greedy on-line Monte Carlo. The assignment comprises of writing a Python-based simulation, and then a report which analyzes and explains the results.

2 Problem Description

In this assignment the agent's goal is to push a bomb into the river as quickly as possible.

1. The robot will be moving on a grid of size $d \times d$.
2. The robot knows its own location and the location of the bomb on the grid.
3. The grid is completely surrounded by a river.
4. The robot can move in a usual manner $\{north, south, east, west\}$
5. The robot can only push the bomb forward. That is, when the robot moves onto the cell the bomb is on, it pushes it one cell forward (depending on the direction the robot came from).
6. The robot's starting position, and the bomb's starting position are random.
7. If the robot itself falls into the river he simply moves back to its previous position.

3 Solutions Required

You are required to solve this task (find the optimal policy) using the following methods:

1. Using Monte-Carlo with a constant α in the update
2. Using Q-learning as discussed in class

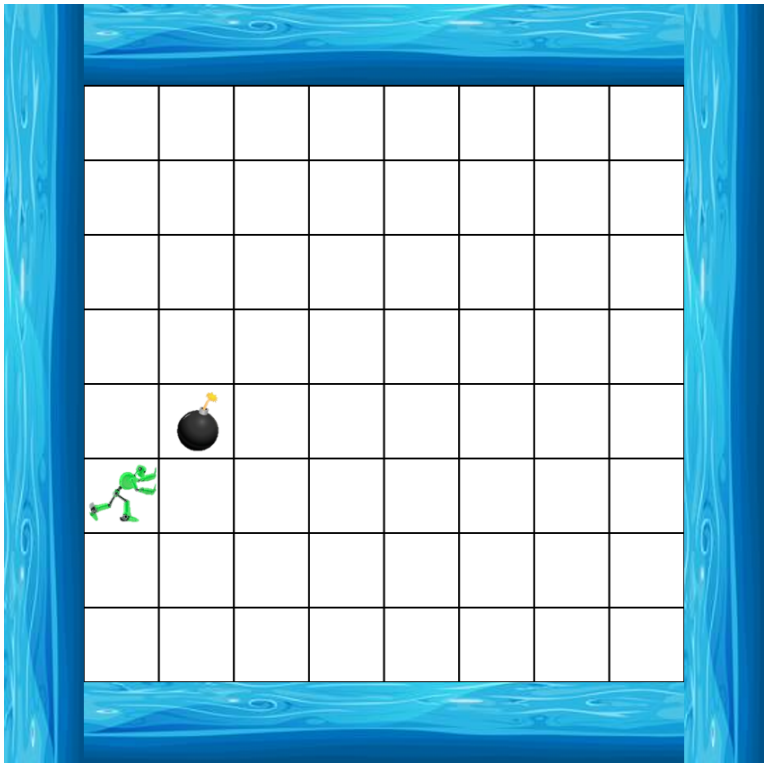

Figure 1: An example setup for the project.

4 Example Setup

For the report you will be required to show results on the specific setup:

1. An 8x8 grid
2. $\alpha = 0.1, \epsilon = 0.1$
3. No discount

You should provide results for two different reward structures:

1. Every step gives a -1 reward
2. Every step gives a -1 reward, moving the bomb farther from the center given a +1 reward, and moving it out gives a +10 reward.

Tip: As some methods can take a long time, it may make sense to limit learning to only episodes under a given number of moves -for example 1000)

5 Coding

Your program should run with the following parameters:

```
python3 myProgram.py d a e n m
```

Where d is the dimension of the grid, a is the value for α, e is the value for ϵ, n is the total number of episodes to train on, and m is the method you are using (1 for MC, 2 for Q-Learning).
You code should include the following functions:

1. A function which calculates the reward given a state and action.
2. A function which performs one episode of Q-learning. This function should return the total return for that episode. It should accept as an argument if it is to use an ϵ-greedy or uniform behavior policy.
3. A function which preforms one episode of Monte Carlo. This function should return the total return for that episode.
4. A function which runs the learning. This function should call either the Q-learning or MC method multiple times (sent as an argument) and store the returns in an array of returns so it can be plotted.
5. A function which plots an episode given a starting state. This will allow you to view episodes after training. An example of an output is shown in figure 2.

You are required to write your code in python3. The only external library you are allowed to use in numpy and matplotlib. You can use internal libraries such as: os, random, timeit, etc. Make sure to comment your code for clarity.

6 Report

Your report should include the following:

1. A short introduction to the problem you are trying to solve.
   Make sure to describe the problem clearly, and how you are planning to solve it.
2. A description of how you framed the problem as an MDP.
   State space, action space and reward structure
3. A description of your code design and your choice of data structures.
   Describe the general structure of your code including functions, classes, and data structures used.
4. Correctness Results using Q learning - in this section you should present the results of your algorithm for the example setup in Sec. 4.
   (a) Show the Q values for this initial setup using your learned Q function.
   (b) Show the path your agent takes in this setup up until the end of an episode which starts the way Fig. 1 shows.
   (c) Repeat a,b for the different reward structures.

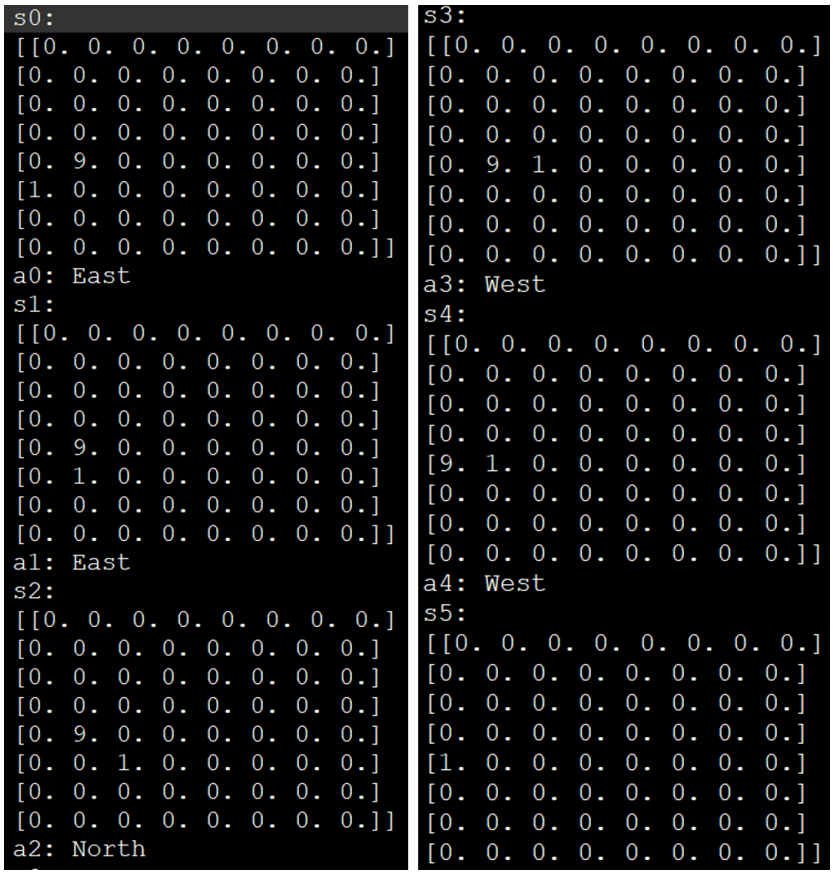

Figure 2: An example output from the project (1-agent, 9-bomb).

(d) Compare how fast the learning happens by plotting the return per episode during learning comparing the different methods (Q-learning vs. Monte Carlo, different reward structures). (similar to the graph presented for the cliff walking in Example 6.6 in the book).

(e) Compare actual leaning time (in seconds) for the different methods.

Besides simply presenting the results discuss them in the text. Why are these results sensible?

5. Conclusion - What have you achieved in this project? What have you learned?

6. References - If necessary.

7 Submission

You are required to submit one zip file with both he report and your code.