Assignment 4: Reinforcement Learning
Machine Learning

Deadline: Sunday 17 Dec 2017, 21:00

Introduction

In this assignment, you will implement Policy Iteration and Q-Learning for a simple Gridworld. Please provide a latex based report in the PDF format in addition to your Python 3 code. You have to use the provided skeleton code and implement the specific functions without changing their names or parameters. Your report and code must be archived in a file named firstname.lastname and uploaded to the iCorsi website before the deadline expires. **Late submissions will result in 0 points.**

Where to get help

We encourage you to use the tutorials to ask questions or to discuss exercises with other students. However, do not look at any report written by others or share your report with others. Violation of that rule will result in 0 points for all students involved. You can contact your TAs using the emails *imanol@idsia.ch* and *boris@idsia.ch*.

Grading

The assignment consists of two tasks totalling at 100 points.

Presets

Consider the task of an agent maneuvering to a goal state in a maze as defined in Figure 1. The agent has four possible actions: North, South, East, West. When the agent takes an action, it moves to the adjacent state in the chosen direction with probability 0.70, and in one of the other directions with probability 0.30 spread evenly. For example, if the agent chooses North, then there is a 70% chance that it actually moves North, a 10% chance it will move South, 10% it will move West, and 10% it will move East. If the agent moves in a direction that will take it outside the maze (e.g. moving South in **S**), it stays in the same state. The reward r is 0 for all state transitions, except that when entering the goal state **G** the reward is 10.0. The discount factor γ is set to 0.9. The agent cannot leave the goal state.

In the skeleton code we use a matrix MAZE to define the maze, such that MAZE(x, y) yields some numerical representation of your world (i.e. $MAZE(x, y) = 0$ if empty, $MAZE(x, y) = 1$ if wall, and $MAZE(x, y) = 2$ if goal state). States are defined by a tuple of each combination of x and y coordinates. Actions are labelled as: N for "North", S for "South", E for "East", and W for "West". Please use a matrix to maintain the value $V(s)$ for each state and a cube to maintain the action-value for each state-action pair $Q(s, a)$.

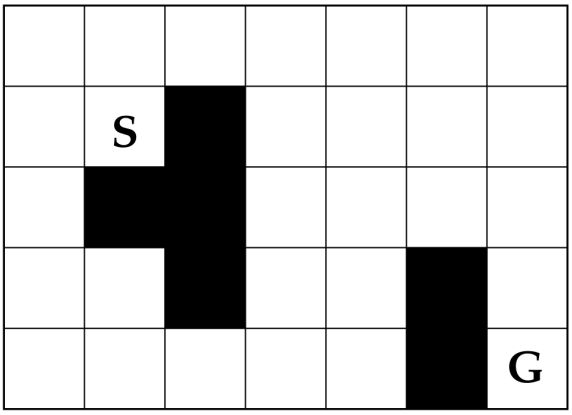

Figure 1: A 5×7 maze, with starting state S and goal state G.

Tasks

1. (a) **(40 points)** Implement the functions in the skeleton code relevant for **Policy Iteration**. Start with $V(s) = 0, \forall s$, a random policy ($\pi(s, a) = 1/4, \forall s, a$), and a stopping criterion where no state has a difference which is bigger than $\epsilon = 1\text{e}{-9}$. What are the final values $V(s)$ after the policy has converged?

   (b) **(10 points)** How and why do the values change if the discount factor γ is changed to 0.7 (again starting with $V(s) = 0, \forall s$)?

2. (a) **(40 points)** Implement the functions in the skeleton code relevant for **Q-learning**. Initialize $Q(s, a) = 0, \forall s, a$ and set the learning rate $\alpha = 0.4$. Starting each episode in state **S**, run Q-learning until it converges, using an ϵ-greedy policy. Each episode ends after 100 actions or once the goal **G** has been reached, whichever happens first.

   (b) **(10 points)** Plot the accumulated reward for the run, i.e. plot the total amount of reward received so far against the number of episodes (especially the first 100 episodes), and plot the final greedy policy.
