DS-GA 3001.005
NYU Center for Data Science

Reinforcement Learning

Homework 02

Exercice 1 (20 points)

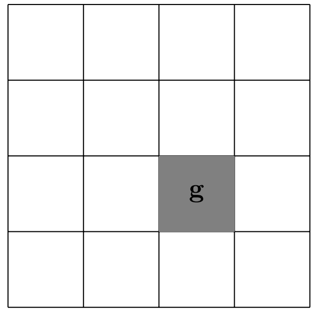
Markov Decision Processes and Bellman Equations:

- **1.1 (10 points)**: What is a Markov Decision Process, and what is the Markov property?
- **1.2 (5 points)**: Write $v_\pi(s)$ as a function of $q_\pi(s, a)$
- **1.3 (5 points)**: Write $v_*(s)$ as a function of $q_*(s, a)$

Exercice 2 (30 points)

Dynamic Programming:

- **2.1 (20 points)**: In the gridworld problem below, the goal is to reach state g, the reward is -1 for moving to any state except state g where it is 0, actions in each state are up, down, right or left (by 1 step), and actions taking the agent off the grid leaves the state unchanged. What are the final state values after convergence of the *Value Iteration* algorithm?



- **2.2 (5 points)**: What is the key difference between the Policy Iteration algorithm and the Value Iteration algorithm?
- **2.3 (5 points)**: What is the key difference between synchronous Value Iteration and asynchronous Value Iteration?

Exercice 3 (25 points)

Monte Carlo and Temporal Difference:

- **3.1 (5 points)**: What is the key difference between Dynamic Programming and sample-based Reinforcement Learning?
- **3.2 (5 points)**: List at least three differences between Monte-Carlo policy evaluation and Temporal Difference policy evaluation
- **3.3 (10 points)**: Why is Q-learning considered an off-policy control method? Provide an example where using an off-policy method is more appropriate than using an on-policy method.
- **3.4 (5 points)**: Why is Q-learning overly optimistic? And how does Double Q-learning help address this issue?

Exercice 4 (25 points)

Produce the code to do the following::

- **4.1 (10 points)**: Write a function that implements the ϵ-greedy action-selection strategy. It should return the action selected. The required input to this function should be listed as arguments and/or defined in the function's docstring.
- **4.2 (15 points)**: Write a function that implements a q-value update for tabular Q-learning, given a reward r observed after sampling an action a in a state s. It should return the updated q-value. The required input to this function should be listed as arguments and/or defined in the function's docstring.