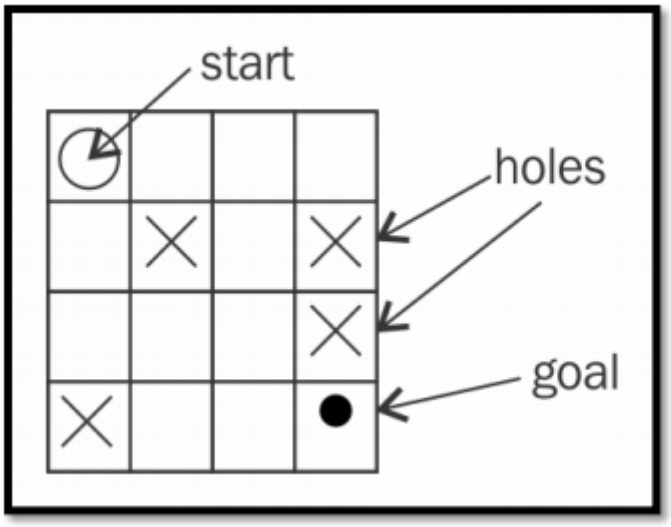
CT5130-34: Reinforcement Learning – Assignment

The “deterministic” FrozenLake is a toy problem from the so called “grid world” category of problems. In this problem the agent lives in a square grid and can move in 4 directions, “up”, “down”, “left” and “right”. The agent always starts in the top-left position and its goal is to reach the bottom right position on the grid (see image below)



Just like the basic Gridworld actions are deterministic i.e. a move to the “right” will always move the agent to the cell directly to their right. The FrozenLake however does have holes in the ice and if the agent falls in, it will drown. Any action which causes a move off the grid results in the agent’s state remaining unchanged.

Assignment:

Feel free to recycle the code for the Gridworld problem from the in-class exercises or create your own from scratch.

Part 1: Create the FrozenLake.

a) Using Python, create a 5x5 grid sized FrozenLake, with a start state at the top left corner and a goal state at the bottom right corner.

b) Place four holes at the following grid positions in the FrozenLake. **(1,0), (1,3), (3,1), (4,2)**

a.

Start (0,0)				
Hole (1,0)			Hole (1,3)	
	Hole (3,1)			
		Hole (4,2)		Goal(4,4)

c) The reward for reaching the goal state is +**10.0**. The reward for falling into a hole is **-5.0 (because you die!)** and the rewards for each transition to a non-terminal state is **-1.0**.

d) The episode ends if the agent falls into a hole or reaches the goal state.

e) The actions are “**up**”, “**down**”, “**left**” and “**right**”.

Part 2: Implement the Reinforcement Learning algorithm Q-learning

a) Using the algorithmic steps outlined in the notes (Week 9 – Model Free Learning (Temporal Difference Learning)) (or S&B book), implement Q-learning on the FrozenLake problem to learn a policy which can navigate optimally through the lake.
b) Set the parameters Alpha = **0.5**, Gamma = **0.9**, and Epsilon = **0.10**
c) Run the frozen lake experiment for **10000** episodes and output the action value estimates at the end of the learning process.
d) Plot a curve of the reward per episode (similar to what was depicted in the slides for the cliff walking task Q-learning vs SARSA).

Submission Instructions

You are free to use Jupyter Notebook

OR

Create a zip archive and submit the following on Blackboard

1. The python source code for your assignment.
2. A Word doc containing the output action value estimates (part 2(c)) for each cell on the grid, similar to the in class exercises only display the max Q(s,a) value at any given cell.
3. Paste in the graph for part 2(d) in the document also.