CSE 6369 - *Reinforcement Learning*

Homework 2- Spring 2020

Due Date: Apr. 21 2020

Dyna-Q

1. Consider the following discretized (Grid World) navigation problem where an agent is moving on a $n \times m$ grid with the actions ”forward”, ”backward”, ”turn left”, and ”turn right” (thus the state representation has to include the orientation of the agent). The agent has unreliable actions where for ”forward” and ”backward” the probability that the agent actually moves to the intended grid cell (in the front or in the back, respectively) is 0.8 and the likelihood that the agent stays in the same cell is 0.2. Similarly, the turn actions turn the agent in the intended direction with a probability of 0.9 and stay in the same orientation with a probability of 0.1. The agent has the ability to detect its location and orientation (and thus the system is Markov in terms of its location and orientation, given fixed obstacles and goal locations).

   The environment contains a number of obstacles as well as a goal, both of which are unknown to the agent. If the agent's action would have it enter an obstacle cell (or leave the grid), the agent will stay in place (i.e. obstacles can not be traversed) and will receive a reward of -100. If the agent reaches the goal, the trial ends and the agent receives a reward of $+100$. All other rewards are 0 (i.e. there is no explicit cost to taking an action).

   The objective of the agent is to learn a policy that allows the agent to reach the goal location from a given start location as efficiently as possible (i.e. with the highest reward possible), given that it initially does not know the transition probabilities and rewards.

   a) Design the state space and MDP for the problem, as well as a model representation for a model-based Dyna-Q style learner.
   b) Implement a Dyna-Q learner for the problem on a 15×25 world.
   c) Run your learner from part $b)$ for three different numbers of offline (i.e. model-based) steps per online step and compare the performance in terms of the number of ”real-world” steps that the learner has to perform to obtain a particular task performance.

Prioritized Sweeping

2. Consider the same task as in part 1) and integrate different methods for picking model-based steps.

   a) Design and implement a priority-based system of picking off-line steps that uses the Bellman error (i.e. implement prioritized sweeping).
   b) Repeat the experiments from part 1c) and compare the results with the ones obtained using the random picks for off-line actions used in Dyna-Q.
   c) Implement a trajectory-based criterion for picking off-line steps and again compare it for the cases from part 1c) with the results obtained in 2b).