HW 1: MDPs, Policy Iteration, and Value Iteration

EECE 571N – Sequential Decision-Making (EECE 571N)

Instructor: Cyrus Neary

Due: **2025-09-29** at 23:59 PT

Instructions. Submit a single PDF to Canvas. Please include your name and student number at the top of that PDF. For all questions, show your work and clearly justify your steps and thinking. Please feel free to include any code as an attachment at the end of the PDF. State any assumptions. Unless otherwise specified, you may collaborate conceptually but must write up your own solutions independently.

Grading. Points for each part are indicated. The total number of achievable points is 100. Partial credit is available for incorrect answers with clear reasoning.

1. **Value functions and optimal value functions.** [10]

   In words, what does $V^\pi(s)$ represent? What does $V^*(s)$ represent?

   Write Bellman's equation for $V^\pi(s)$, and Bellman's optimality equation for $V^*(s)$.

2. **T_π is a γ-contraction in $||\cdot||_\infty$.** [15]

   Recall the definition of the Bellman operator T_π for policy π from class.

$$(T_\pi V)(s) = \sum_{a \in A} \pi(a \mid s) \left(R(s,a) + \gamma \sum_{s' \in S} T(s' \mid s,a) V(s') \right)$$

   That is, T_π is an operator that maps value functions $V(s)$ to new value functions $V'(s) = (T_\pi V)(s)$.

   Show that T_π is a γ-contraction in $||\cdot||_\infty$. That is, show that

$$||T_\pi V - T_\pi W||_\infty \leq \gamma ||V - W||_\infty, \quad \forall V, W.$$

3. **Banach fixed point theorem.** [5]

   State Banach's fixed point theorem (Hint: Wikipedia).

4. **Convergence of policy evaluation by iteration of the Bellman operator.** [10]

   Show that for any initial value function V_0, the sequence defined by $V_{n+1} = T_\pi V_n$ converges to the value function V^π for policy π (Hint: Use the results from the last two questions).

5. **Formalizing a gridworld MDP.** [20]

   Define all of the components of an MDP $\mathcal{M} = (S, A, T, \gamma, R, \mu)$ for the gridworld environment we saw in class (illustrated in Figure 1).

   (a) Assume the gridworld is surrounded by impassable walls (if the agent takes an action that results moving into the wall, they should instead remain in place).

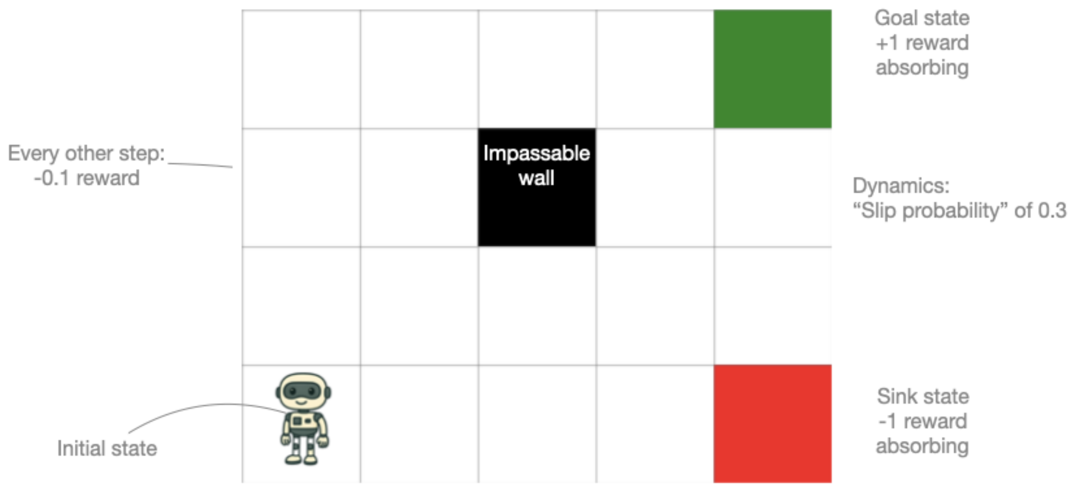

Figure 1: Example gridworld from class.

(b) Assume the goal state and the sink state are absorbing (once the agent transitions into them, they should stay there with probability 1).

(c) Define the transition dynamics to include a slip probability of p. i.e., with probability $1 - p$ the agent will move in their intended direction, and with probability p they will transition to a random neighboring square.

(d) Define the reward function to reward the agent with $+1$ if they arrive at the goal location, -1 if they arrive at the sink location, 0.0 if they are already at either the goal or sink location, and -0.1 for every other step. In class, we've spoken only of reward functions $R(s, a)$ depending on state s and action a. How can we express the reward for this problem in the form $R(s, a)$ instead of $R(s, a, s')$?

6. **Implement the MDP in Python.** [30]

- Use Python to define the state and action spaces, and to implement the transition function $T(s' \mid s, a)$ and the reward function $R(s, a)$.

  Hint: It will be very helpful for the next questions if you implement a function MDP.step(s, a) that takes the current state s and action a as input, and outputs a list $out = [outcome_1, outcome_2, ...outcome_k]$ of ALL k possible outcomes for that action. Each outcome should contain the following information: $outcome = (prob, s', r)$ the next state s', the reward r, and the probability $prob$ of the transition.
- Implement a uniform policy as an array of size $S \times A$ with policy$[s, a] = \pi(a \mid s)$.
- Implement the iterative policy evaluation algorithm from class to compute $V^\pi(s)$. Plot the value function (either as a heatmap or as a grid of values for each state). Hint: Use the MDP.step(s,a) function that you previously implemented.

7. **Implement the value iteration algorithm.** [10]

Implement the value iteration algorithm discussed in class for your MDP. Hint: First implement the optimal bellman operator, then implement the iterative value iteration procedure. Include the optimal value function at convergence (either as a heatmap or as a grid of values).