

# ASEN 5264 Decision Making under Uncertainty

## Homework 3: Online MDP Methods

February 9, 2026

A submission should consist of two or three files:

- A single PDF file containing answers to questions (typed, handwritten, or exported notebook).
- JSON file output from `HW3.evaluate`.
- A code listing *if the code is not included in the PDF*.

### 1 Conceptual Questions

**Question 1.** (10 pts)

- a) Suppose that the optimal value function  $U^*$  for an MDP with discrete state and action spaces is known. Write an equation in terms of  $(S, A, T, R, \gamma)$  for extracting the optimal policy  $\pi^*$  from  $U^*$ .

- b) Suppose you have an MDP defined by

$$S = [0, \infty), \quad A = \{0, 0.1, 1\}, \quad \gamma = 1.0$$

$$T(s'|s, a) = \mathcal{U}([s + a, s + a + 1])$$

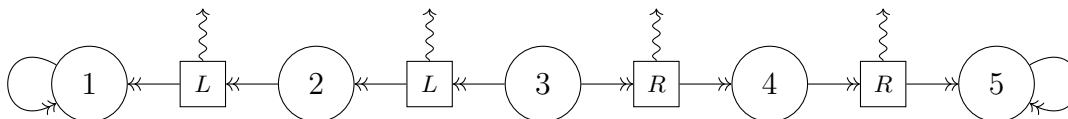
$$R(s, a) = -\mathbb{1}(s \leq 1) - a$$

and a policy  $\tilde{\pi}$  with  $U^{\tilde{\pi}}(s) = -2 \cdot \mathbb{1}(s \leq 1)$ . If the current state is 0.8, which action maximizes  $Q^{\tilde{\pi}}(s, a)$ ? Justify your answer.

**Question 2.** (10 pts) Do similar  $Q$  values imply similar rewards? Consider the following claim:

If a policy  $\pi$  satisfies  $|Q^*(s, \pi^*(s)) - Q^*(s, \pi(s))| \leq \beta$  for all  $s \in S$  for some  $\beta > 0$ , then it immediately follows that  $|R(s, \pi^*(s)) - R(s, \pi(s))| \leq \beta$  for any  $s \in S$ .

It turns out that this claim is incorrect.<sup>1</sup> In this exercise, you will formulate a counterexample demonstrating that it is false. Consider the MDP below:



The state space is  $S = \{1, \dots, 5\}$  and the action space is  $A = \{L, R\}$  (but not all actions are available from each state). Transitions are deterministic as shown. The discount factor is  $\gamma = 0.9$ .

Choose a reward function,  $R$ , (i.e. values for the squiggly arrows), a policy,  $\pi$ , and a value  $\beta$  that constitute a counterexample to the claim above.<sup>2</sup> Justify your answer.

<sup>1</sup>Even seasoned researchers can be tripped up by this - this claim was erroneously made in the proof for Lemma 5 of the Sparse Sampling paper by Kearns, Mansour, and Ng <https://www.cis.upenn.edu/~mkearns/papers/sparsesampling-journal.pdf>.

<sup>2</sup>To demonstrate that you have found a counterexample, use the following steps: (1) Choose  $R$ ,  $\pi$  and  $\beta$  (note that to choose  $\pi$ , you only have to choose  $\pi(3)$  because all other actions are pre-determined). (2) Verify that  $Q^*(s, \pi^*(s))$  and  $Q^*(s, \pi(s))$  are closer than  $\beta$  for all states. (3) Find one state where the difference between  $R(s, \pi^*(s))$  and  $R(s, \pi(s))$  is greater than  $\beta$ . (4) If it is not possible, then revise  $R$ ,  $\pi$ , and  $\beta$  and try again.

## 2 Exercises

HW3.DenseGridWorld() generates a 60x60 grid world MDP. There is a reward of +100 every 20 cells, i.e. at [20,20], [20,40], [40,20], etc. After the agent reaches one of these reward cells, the problem terminates. All cells also have a cost. Only a generative transition model is available. You will use the following functions from POMDPs.jl to interact with this problem (or larger versions) in the rest of this assignment:

- actions(m)
- @gen(:sp, :r)(m, s, a)
- isterminal(m, s)
- discount(m)
- statetype(m)
- actiontype(m)

### Question 3. (15 pts) Monte Carlo Policy Evaluation

a) Write a rollout simulation function for an MDP starting with the following code:

```
r_total = 0.0
t = 0
while !isterminal(mdp, s) && t < max_steps
    a = :down # replace this with a policy
    s, r = @gen(:sp,:r)(mdp, s, a)
    r_total += discount(m)^t*r
    t += 1
end
```

Use this function to perform a Monte Carlo evaluation of a uniform random policy on an MDP created with HW3.DenseGridWorld(seed=3). Report the mean discounted reward estimate and standard error of the mean (SEM). Run enough simulations so that the SEM is less than 5.

b) Create a heuristic policy that improves upon the random policy by at least 50 reward units. Report the mean and standard error from a Monte Carlo evaluation.

### Question 4. (20 pts) Monte Carlo Tree Search

Write code that performs 7 iterations of Monte Carlo Tree Search on an MDP created with HW3.DenseGridWorld(seed=4), starting at state (19,19). You will need to produce three dictionaries:

- Q maps  $(s, a)$  tuples to Q value estimates.
- N maps  $(s, a)$  tuples to N, the number of times the node has been tried.
- t maps  $(s, a, s')$  tuples to the number of times that transition was generated during construction of the tree.

Then visualize the resulting tree with HW3.visualize\_tree(Q, N, t, SA[19, 19])<sup>3</sup>. **Submit an image of the tree, the code used to generate it, and a few sentences describing the tree after 7 iterations** (e.g. which actions have the highest Q values? Does this make sense?).

### Question 5. (15 pts) Planning with MCTS

Use your Monte Carlo tree search from Question 4 to plan online in the simulation loop. Use 1000 iterations of MCTS to choose each action. Evaluate the MCTS planner with 100 100-step Monte Carlo simulations. Report the mean accumulated reward and standard error of the mean.

---

<sup>3</sup>SA is from the StaticArrays.jl package.

### 3 Challenge Problem

**Question 6.** (10 pts code and description, 20 pts score) Fast Online Planning

Create a function `select_action(m,s)` that takes in a  $100 \times 100$  `DenseGridWorld`, `m`, and a state `s`, and returns a near-optimal action. You may wish to base this code on the MCTS code that you wrote for Question 4. Evaluate this function with `HW3.evaluate` and **submit the resulting json file along with the code and a one paragraph to one page description of your approach**, including tuning parameters that worked well, the rollout policy, etc. A score of 50 will receive full credit. In order to achieve a score above 50, you will be limited to 50ms of planning time per step. There are no restrictions on this problem - you may wish to use a different algorithm, multithreading, etc. Starter code on github will give suggestions for timing and other details.