Assignment5 – Markov Decision Procedures

Given: May 30 Due: June 3

Problem 5.1 (HMMs in Python)

Implement filtering, *prediction* and *smoothing* for *HMMs* in Python by completing the *implementation* of `hmm.py` at `https://kwarc.info/teaching/AI/resources/AI2/hmm/`.

Hint: This problem uses `numpy`, which is a Python *library* for working with arrays/matrices. If you have never worked with `numpy` before, you can find many high-quality introductions online. Due to its popularity and frequent use for machine learning etc., it is definitely worth getting to know `numpy`. That being said, you only need very few and basic numpy functions for this assignment, which you should be able to find without problems (e.g. searching for *numpy matrix multiplication*).

Solution: See `https://kwarc.info/teaching/AI/resources/AI2/hmm/`.

Problem 5.2 (Markov Decision Processes)

1. Give an optimal policy π^* for the following MDP:
   - set of states: $S = \{0, 1, 2, 3, 4, 5\}$ with initial state 0
   - set of actions for $s \in S$: $A(s) = \{-1, 1\}$
   - transition model for $s, s' \in S$ and $a \in A(s)$: $P(s' \mid s, a)$ is such that
     - $s' = (s + a) \mod 6$ with probability $2/3$,
     - $s' = (s + 3) \mod 6$ with probability $1/3$.
   - reward function: $R(5) = 1$ and $R(s) = -0.1$ for $s \in S \setminus \{5\}$

   Solution: $\pi^*(s) = 1$ if $s \in \{3, 4\}$ and $\pi^*(s) = -1$ if $s \in \{0, 1\}$ and arbitrary for $s \in \{2, 5\}$

   ---

2. State the Bellman equation.

   Solution: $U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s' \in S} U(s')P(s' \mid s, a)$

   ---

3. Complete the following high-level description of the value iteration algorithm:

- The algorithm keeps a table $U(s)$ for $s \in S$, that is initialized with

- In each iteration, it uses the

  in order to

- $U(s)$ will converge to the

Solution:

- The algorithm keeps a table $U(s)$ for $s \in S$, that is initialized with arbitrary values, e.g. all 0 or the rewards.
- In each iteration, it uses the Bellman equation in order to update $U(s)$.
- $U(s)$ will converge to the expected utility of s.

1 per blank; in each case 0.5 if mistake

Problem 5.3 (Bellman Equation)

State the *Bellman equation* and explain every symbol in the equation and what the equation is used for and how.

Solution:

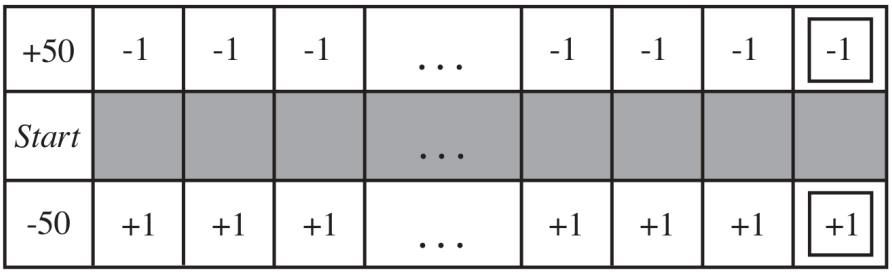
$$U(s) = R(s) + \gamma \cdot \max_{a \in A(s)} \left(\sum_{s'} P(s' \mid s, a) \cdot U(s') \right)$$

The meaning of the components is as follows:

- $U(s)$: the utility of the state s (long-term, global)
- $R(s)$: the reward at state s (short-term, local)
- $A(s)$: the set of actions available in state s
- $\max_{a \in A(s)}$: take the maximum over all available actions in state s
- $P(s' \mid s, a)$: the probability that taking action a in state s yields state s'
- $U(s')$: the utility in successor state s'
- $\left(\sum_{s'} P(s' \mid s, a) \cdot U(s')\right)$: the expected utility of action a by summing over all possible successor states

The equation is used to compute the utility of every state. The algorithm uses the equation as an iteration operator that computes new values for every $U(s)$ by evaluating the right hand side for the current values of U. If this leads to a *fixpoint*, a solution for the utilities has been found.

Problem 5.4 (MDP Example)
Consider the following world:



The world is 101 fields wide (i.e., 203 fields in total). In the $Start$ state an agent has two possible actions, Up and $Down$. It cannot return to $Start$ though and the cannot pass gray fields, so after the first move the only possible action is $Right$.

1. Model this world as a Markov Decision Process, i.e., give the components S, s_0, A, P, and R.

Solution:

- Set of states $S = \{-101, \dots, 0, \dots, 101\}$.
  Note that the set of states can be swapped out arbitrarily against any other set of the same size. The choice made is practical because it allows using 0 as the start state and n as the state $|n|$ steps away from the start.
- Initial state: $s_0 = 0$.
- Reward function R: $R(0) = 0$, $R(1) = 50$, $R(-1) = -50$, $R(s) = -1$ for $s \in \{2, \dots, 101\}$, $R(s) = 1$ for $s \in \{-2, \dots, -101\}$
- Possible actions in each state: $A(0) = \{Up, Down\}$, $A(n) = \{Right\}$ for all $n \in S \setminus \{0\}$
- Transition model $P(s' \mid s, a)$: This world is deterministic — the successor state of each action is uniquely determined. Therefore, all probabilities are either 1 or 0.
  - current state $s = 0$: $P(1 \mid 0, Up) = 1$, $P(1 \mid 0, Down) = 0$, $P(-1 \mid 0, Up) = 0$, $P(-1 \mid 0, Down) = 1$
    and $P(s' \mid 0, a)$ for all $s' \in S \setminus \{-1, 0, 1\}$ and $a \in \{Up, Down\}$
  - current state $s \neq 0$:

* $P(s+1 \mid s, Right) = 1$ for $s \in \{1, \ldots, 100\}$, $P(101 \mid 101, Right) = 1$
* $P(s-1 \mid s, Right) = 1$ for $s \in \{-1, \ldots, -100\}$, $P(-101 \mid -101, Right) = 1$

All other probabilities are 0.

---

2. For what discount factors γ should the agent choose Up and for which $Down$? Compute the utility of each action (i.e., the utility of the successor state) as a function of γ.

Solution: We have $U(s) = R(s) + \gamma \max_a \left(\sum_{s'} U(s')\right)$, since all transitions are deterministic. Then

$$U(1) = 50 + \gamma(-1 + \gamma(-1 + \ldots)) = 50 - \sum_{i=1}^{100} \gamma^i$$

$$U(-1) = -50 + \gamma(1 + \gamma(1 + \ldots)) = -50 + \sum_{i=1}^{100} \gamma^i$$

and for $i > 0$:

$$U(i) = \sum_{k=1}^{i} \gamma^k \qquad U(-i) = -\sum_{k=1} \gamma^k$$

So we need to solve the following equation for γ:

$$50 - \sum_{i=1}^{100} \gamma^i = -50 + \sum_{i=1}^{100} \gamma^i$$

$$50 = \sum_{i=1}^{100} \gamma^i$$

We get $\gamma \approx 0.984397669$, and we should go Up if γ is smaller.

---

3. What is the optimal policy if the upper path is better?

Solution: The optimal policy π^* maps each $s \in S$ to an element of $A(s)$. Because most states have only one action, we immediately have $\pi^*(s) = Right$ for $s \neq 0$. For $s = 0$, we have $\pi^*(0) = Up$.
