Assignment5 – Markov Decision Procedures

Given: May 30   Due: June 9

Problem 5.1 (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 $s \in S \setminus \{5\}$
2. State the Bellman equation.
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

Problem 5.2 (Bellman Equation)

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

Problem 5.3 (MDP Example)

Consider the following world:

+50	-1	-1	-1	. . .	-1	-1	-1	$\boxed{-1}$
Start				. . .				
-50	+1	+1	+1	. . .	+1	+1	+1	$\boxed{+1}$

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

Problem 5.4 (Value Iteration for Navigation)

Implement value iteration for an agent navigating worlds like the 4x3 world from the lecture notes. The agent has four possible actions: *right*, *up*, *left*, *down*. The probability of actually moving in the intended direction is p and the probability of moving in one of the orthogonal directions is $\frac{1-p}{2}$ respectively. For example, if $p = 0.8$ and the chosen action is *up*, the agent will actually move up with a probability of $p = 0.8$ and will move left and right with a probability of 0.1 each. If the agent ends up moving in a direction that has no free adjacent square, it will remain on its current square instead. For example, if the agent is on square $(0, 0)$ with the action *up*, it will end up on square $(0, 1)$ with a probability of p, on square $(1, 0)$ with a probability of $\frac{1-p}{2}$ and on square $(0, 0)$ with a probability of $\frac{1-p}{2}$.

(0, 2) -0.040 → 0.647	(1, 2) -0.040 → 0.753	(2, 2) -0.040 → 0.855	(3, 2) 1.000 T 1.000
(0, 1) -0.040 ↑ 0.557	(1, 1) W	(2, 1) -0.040 ↑ 0.569	(3, 1) -1.000 T -1.000
(0, 0) -0.040 ↑ 0.465	(1, 0) -0.040 ← 0.386	(2, 0) -0.040 ↑ 0.451	(3, 0) -0.040 ← 0.230

Results for 4x3 world with $p = 0.8$, $\gamma = 0.95$, $\varepsilon = 0.001$.

A skeleton *implementation* with technical instructions can be found at `https://kwarc.info/teaching/AI/resources/AI2/mdp/`. It also allows the visualization of the computed utilities and policy (see figure above): Each square is annotated with the coordinates, the reward, the computed policy and the computed utility. Walls and terminal nodes don't have a policy and are marked with W and T respectively.

Hint: You will also have to compute a policy based on the utilities obtained from value iteration. For that, you should pick the actions that *maximize* the expected utility. A common mistake is the assumption that the best policy is always to go in the direction of the square with the maximal utility.
