Lehrstuhl für STEUERUNGS- UND REGELUNGSTECHNIK Technische Universität München Prof. Dongheui Lee	**MACHINE LEARNING IN ROBOTICS** Exercises 6: Reinforcement Learning

Exercise 1

Consider a robot that lives in a grid world. The robot can execute only four actions i.e. move to North, South, East or West. The G corresponds to a terminal goal state and reaching goal state yields a reward of +10. Similarly D corresponds to a ditch (another terminal state) that should be avoided and reaching ditch yields a negative reward of -20. X represent blocked states that are not accessible. If the robot executes an action that tries to get to the blocked state or out of the grid then it stays at the same position. All other states yield a negative reward of -1 (for encouraging the robot to reach the goal state as quickly as possible). The model of robot is stochastic and with probability 0.8, it moves in the commanded direction but with probability of 0.2 it either moves left or right (0.1 for each).

Using a discount factor of $\gamma = 0.99$, write few steps of value iteration using asynchronous update and use it to get the optimal policy $\pi^*(s)$.

G			D
X	X		
X	X		

Solution Exercise 1

Table 1: Value iteration.

```
For each state $s$, initialize $V(s) := 0$.
Repeat until convergence
{
For every state, update $V(s) := R(s) + \max_{a \in A} \gamma \sum_{s'} P_{sa}(s')V(s')$
}
```

Step 1: Initialize value of all states to zero.

Step 2: Calculate value of each state:

$$V(1,2) = -1 + \max[\underbrace{0.99 * ((0.8)(0) + (0.1)(0) + (0.1)(0))}_{action=East}, \underbrace{0.99 * ((0.8)(10) + (0.1)(0) + (0.1)(0))}_{action=West},$$
$$\underbrace{0.99 * ((0.8)(0) + (0.1)(0) + (0.1)(10))}_{action=North}, \underbrace{0.99 * ((0.8)(0) + (0.1)(0) + (0.1)(10))}_{action=South}]$$
$$= -1 + \max[0, 7.92, 0.99, 0.99] = 6.92$$

$$V(1,3) = -1 + \max[\underbrace{0.99 * ((0.8)(-20) + (0.1)(0) + (0.1)(0))}_{action=East}, \underbrace{0.99 * ((0.8)(6.92) + (0.1)(0) + (0.1)(0))}_{action=West},$$
$$\underbrace{0.99 * ((0.8)(0) + (0.1)(6.92) + (0.1)(-20))}_{action=North}, \underbrace{0.99 * ((0.8)(0) + (0.1)(6.92) + (0.1)(-20))}_{action=South}]$$
$$= -1 + \max[-15.84, 5.4806, -1.2949, -1.2949] = 4.4806$$

$\vdots$

After first iteration:

G	6.92	4.4806	D
X	X	2.5487	-0.7477
X	X	1.0185	-0.26

The procedure is repeated until convergence. At convergance the value of each state is:

G	8.6284	7.0528	D
X	X	5.2613	1.5615
X	X	3.7766	2.3814

The optimal policy $\pi^*(s)$ is defined as:

$$\pi^*(s) = \arg\max_{a \in A} \gamma \sum_{s' \in S} P_{sa}(s')V^*(s')$$

Using the current values the policy can be derived like this.

G	$\leftarrow$	$\leftarrow$	D
X	X	$\uparrow$	$\downarrow$
X	X	$\uparrow$	$\leftarrow$

(For e.g. action west for state $(2,4)$ yields $(0.8)(5.2613) + (0.1)(-20) + (0.1)(2.3814) = 2.4472$, action north yields $(0.8)(-20) + (0.1)(1.5615) + (0.1)(5.2613) = -15.3177$, action east yields $(0.8)(1.5615) + (0.1)(-20) + (0.1)(2.3814) = -0.5127$ while action south yields $(0.8)(2.3814) + (0.1)(1.5615) + (0.1)(5.2613) = 2.5874$. Hence action south is the optimal action for state $(2,4)$.)

Exercise 2

Solve the same problem but now with Policy Iteration.

Solution Example 2

Table 2: Policy iteration.

```
Initialize $\pi$ randomly.
Repeat until convergence
{
(a) Let $V := V^\pi$
(b) For each state s, let $\pi(s) := \arg\max_{a \in A} \sum_{s'} P_{sa}(s')V(s')$
}
```

The first step in policy iteration is to define an initial policy (can be randomly defined):

G	$\leftarrow$	$\leftarrow$	D
X	X	$\leftarrow$	$\leftarrow$
X	X	$\leftarrow$	$\leftarrow$

$V^\pi(s) = R(s) + \gamma \sum\limits_{s' \in S} P_{s\pi(s)}(s')V^\pi(s')$ (Bellman's equation)

Next step is to estimate the value of each state for current policy by using Bellman's equation:

$V(1,2) = -1 + 0.99 * (0.8 * 10 + 0.2 * V(1,2));$

$V(1,3) = -1 + 0.99 * (0.8 * V(1,2) + 0.1 * V(1,3) + 0.1 * V(2,3));$

$V(2,3) = -1 + 0.99 * (0.8 * V(2,3) + 0.1 * V(1,3) + 0.1 * V(3,3));$

$V(2,4) = -1 + 0.99 * (0.8 * V(2,3) + 0.1 * (-20) + 0.1 * V(3,4));$

$V(3,3) = -1 + 0.99 * (0.8 * V(3,3) + 0.1 * V(3,3) + 0.1 * V(2,3));$

$V(3,4) = -1 + 0.99 * (0.8 * V(3,3) + 0.1 * V(2,4) + 0.1 * V(3,4));$

This gives 6 linear equations in six unknowns which can be easily solved to get the value of each state. Solving the above equations we get.

G	8.6284	5.1759	D
X	X	-11.8209	-14.1186
X	X	-19.9108	-17.9435

After that the improved policy can be updated as $\pi(s) := \arg\max_{a \in A} \sum_{s'} P_{sa}(s')V(s')$ (as was done at the last step of value iteration).

G	←	←	D
X	X	↑	←
X	X	↑	↑

Now again we have a fixed policy for which we can claculate the value of each state and then the policy can be updated greedily for the calculated values. The policy evaluation and policy improvement steps continue until the policy converges to optimal policy $\pi^*(s)$.

At convergence we will get to the same policy as was calculated by the Policy Iteration.

G	←	←	D
X	X	↑	↓
X	X	↑	←