Reinforcement Learning	March 27, 2019
Recitation 5	
Lecturer:Yishay Mansour	*TA: Lee Cohen*

So far, we have discussed MDPs and algorithms for MDPs assuming that the state transition probabilities and rewards are known. In many realistic problems, we are not given state transition probabilities and rewards explicitly, but must instead estimate them from data (Usually, S, A and γ are known.). If we don't know P and R (i.e., we know only S and A), then we get **reinforcement learning**.

Definition 1. Reinforcement Learning algorithm categories:

1. **Model-based** where the algorithm first learns the MDP (transition P and rewards R), then learns the policy.
2. **Model-free** Don't learn a model, learn value function or policy directly.
3. **On-policy** Agent learns the value of the policy being used, including exploration actions.
4. **Off-policy** The target policy is learned regardless (independently) of the actions chosen for exploring the environment. The agent follows a policy but learns the value of a different policy.

Part 1: Off-policy Model-Based

Definition 2. Model-based method learns the MDP before computing π_{opt} and V_{opt}. Given an "experience" in the MDP consisting of a number of trajectories:

$$s_0^{(0)}, a_1^{(0)}, r_1^{(0)}, s_1^{(0)}, a_2^{(0)}, r_2^{(0)}, s_2^{(0)}, \ldots, a_n^{(0)}, r_n^{(0)}, s_n^{(0)} \tag{1}$$

$$s_0^{(1)}, a_1^{(1)}, r_1^{(1)}, s_1^{(1)}, a_2^{(1)}, r_2^{(1)}, s_2^{(1)}, \ldots, a_n^{(1)}, r_n^{(1)}, s_n^{(1)} \tag{2}$$

$$\vdots \tag{3}$$

$$s_0^{(m)}, a_1^{(m)}, r_1^{(m)}, s_1^{(m)}, a_2^{(m)}, r_2^{(m)}, s_2^{(m)}, \ldots, a_n^{(m)}, r_n^{(m)}, s_n^{(m)} \tag{4}$$

The P and R of the MDP can be approximated as

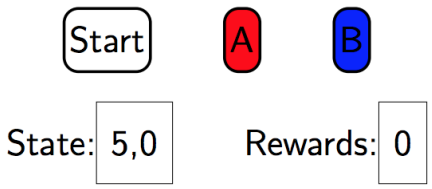
$$\hat{P}(s, a, s') = \frac{count(s, a, s')}{count(s, a)} \tag{5}$$

$$\hat{R}(s,a) = \frac{\sum_{(s,a,r,s')} r}{count(s,a)} \tag{6}$$

If $count(s,a) = 0$ then set $P(s,a,s') = \frac{1}{|S|}$.
Then we can use value iteration or policy iteration to find the optimal policy and values.

Example 1. Consider **Mystery Buttons** game.
For each round $i = 1, 2, ...$:

- You choose button A or B.
- You move to a new state and get some rewards.



Given the following game experience:

$$S1; A, 3, S2; B, 0, S1; A, 5, S1; A, 7, S1 \tag{7}$$

the estimate of of the MDP are:

$$\begin{aligned} \hat{P}(S1, A, S1) &= \frac{0+1+1}{3} = \frac{2}{3} \\ \hat{P}(S1, A, S2) &= \frac{1+0+0}{3} = \frac{1}{3} \\ \hat{R}(S1, A) &= \frac{5+7+3}{3} = 5 \end{aligned} \tag{8}$$

For unseen transitions:

$$\hat{P}(S2, A, *) = \frac{1}{|S|} = \frac{1}{2} \tag{9}$$

Example 2. Initial state error. In class we saw bounds for approximated rewards and approximated transition matrix learned from observed trajectories. We will show that if we are given the transition matrix P and an **approximated** start distribution - then the error is bounded and does not increase over time.
Define:
x - start distribution
$\hat{x}$ - start distribution with error
$x - \hat{x} = z$ - difference vector
α - the maximal error between 2 states i.e. $||z||_\infty \leq \alpha$

P^t - the transition matrix at time t.
zP^t - the difference at time t.
It follows:

$$|(zP^t)(i)| = |\sum_{j=1}^{|S|} z(j)P^t(j,i)| \leq \sum_{j=1}^{|S|} |z(j)||P^t(j,i)| \leq \alpha \sum_{j=1}^{|S|} |P^t(j,i)| \leq \alpha|S| \tag{10}$$

Therefore, the maximal error in a single state is bounded and does not increase with time.

Example 3. Error in transition matrix. Next, we show an example for the case where a small error in the transition function can have large effect as time progresses.
Consider the following 1-dimensional random walk:

$$Z_+ = \begin{cases} 1 & \frac{1}{2} + \epsilon \\ -1 & \frac{1}{2} - \epsilon \end{cases} \tag{11}$$

Let $S_T = \sum_{i=1}^{T} Z_i$ denote the state at time T. The borders of the walk are at $M >> \frac{1}{\epsilon^2}$. The intuition is that the walk will drift to the positives, since the expectation of each step is $E(Z_+) = 2\epsilon$.

Without the error in the transition matrix:

$$Z = \begin{cases} 1 & \frac{1}{2} \\ -1 & \frac{1}{2} \end{cases} \tag{12}$$

there is no drift for any directions because: $E(S_T) = \sum_{i=1}^{T} E(Z_i) = 0$

We will show that when $T >> \frac{1}{\epsilon^2}$, the probability of being in the negative side i.e., $Pr\left[S_T < 0\right]$, is very small.

$$\begin{aligned} Pr\left[S_T < 0\right] = Pr\left[\sum_{i=1}^{T} Z_i < 0\right] = \\ Pr\left[\frac{1}{T}\sum_{i=1}^{T} Z_i < 0\right] = \\ Pr\left[2\epsilon - \frac{1}{T}\sum_{i=1}^{T} Z_i > 2\epsilon\right] \leq \\ Pr\left[|2\epsilon - \frac{1}{T}\sum_{i=1}^{T} Z_i| > 2\epsilon\right] \end{aligned} \tag{13}$$

Recall Chernoff bound, given X_i independent identically distribution bounded random variables, with $a \leq X_i \leq b$ for all i then:

$$Pr[|E(X) - \frac{1}{m}\sum_{i=1}^{m} X_i| > \gamma] \leq 2exp(-\frac{2m\gamma^2}{(b-a)^2}) \tag{14}$$

Plugging into our inequality:

$$Pr\left[|2\epsilon - \frac{1}{T}\sum_{i=1}^{T} Z_i| > 2\epsilon\right] \leq 2exp(-\frac{2T(2\epsilon)^2}{(1-(-1))^2}) = 2exp(-2T\epsilon^2) \tag{15}$$

For large enough T i.e., $T = \frac{100}{\epsilon^2}$, we will get $Pr[S_T < 0]$ small.

For the symmetric case,

$$Z_- = \begin{cases} 1 & \frac{1}{2} - \epsilon \\ -1 & \frac{1}{2} + \epsilon, \end{cases} \tag{16}$$

it can be shown in the same way that a negative drift occurs for large enough T.

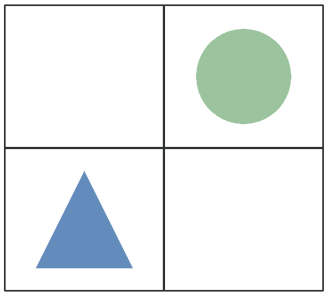
Part 2: On-policy Model-Based

Rmax Algorithm

Input: $\mathbf{S}$, $\mathbf{A}$, γ, m, Rmax
Output: π^*

Initialize counter $c(s,a) \leftarrow 0$ for all $(s,a) \in \mathbf{S} \times \mathbf{A}$
Initialize the empirical known state-action MDP $\hat{M} = \langle \mathbf{S}, \mathbf{A}, \hat{\mathbf{T}}, \hat{\mathbf{R}}, \gamma \rangle$:
    $\hat{T}(s,a,s') = \mathbb{I}(s'=s), \quad \hat{R}(s,a) = Rmax$
for $t = 1, 2, 3, \ldots$ **do**
    Compute $\hat{\pi}_t^*$
    Execute $a_t = \hat{\pi}_t^*(s_t)$
    Act: receive reward r_t and transition to the next state s_{t+1}.
    $c(s_t, a_t) \leftarrow c(s_t, a_t) + 1$
    if $c(s_t, a_t) = m$ **then**
        Redefine: $\hat{T}(s,a,s') = \frac{count(s,a,s')}{count(s,a)}$ and $\hat{R}(s,a) = \frac{\sum_{(s,a,r,s')} r}{count(s,a)}$

Example 4. Consider the following 2×2 grid world. The agent starts at location $(1,1)$ and the goal is at (2,2). The agent can move [left, right, down, up]. The reward is 0, unless we step into a goal state and then it is 1.



We will use Rmax to train an agent to solve the game with the following inputs:
$S = \{(1,1),(1,2),(2,1),(2,2)\}$,$A = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$,$\gamma = 0.99$,$m = 2$.

init:

Table 1: Known status, Reward counts, Transition counts

	(1,1)	(1,2)	(2,1)	(2,2)
↑	U	U	U	U
→	U	U	U	U
↓	U	U	U	U
←	U	U	U	U

	(1,1)	(1,2)	(2,1)	(2,2)
↑	Rmax	Rmax	Rmax	Rmax
→	Rmax	Rmax	Rmax	Rmax
↓	Rmax	Rmax	Rmax	Rmax
←	Rmax	Rmax	Rmax	Rmax

	(1,1)	(1,2)	(2,1)	(2,2)
↑	0	0	0	0
→	0	0	0	0
↓	0	0	0	0
←	0	0	0	0

t=1: Agent moves left (s: (1,1) - action chosen in step 4 & 5)

Table 2: Known status, Reward counts, Transition counts

	(1,1)	(1,2)	(2,1)	(2,2)
↑	U	U	U	U
→	U	U	U	U
↓	U	U	U	U
←	U	U	U	U

	(1,1)	(1,2)	(2,1)	(2,2)
↑	Rmax	Rmax	Rmax	Rmax
→	Rmax	Rmax	Rmax	Rmax
↓	Rmax	Rmax	Rmax	Rmax
←	Rmax [0]	Rmax	Rmax	Rmax

	(1,1)	(1,2)	(2,1)	(2,2)
↑	0	0	0	0
→	0	0	0	0
↓	0	0	0	0
←	1	0	0	0

t=2: Agent moves left (s: (1,1))

Table 3: Known status, Reward counts, Transition counts

	(1,1)	(1,2)	(2,1)	(2,2)
↑	U	U	U	U
→	U	U	U	U
↓	U	U	U	U
←	K	U	U	U

	(1,1)	(1,2)	(2,1)	(2,2)
↑	Rmax	Rmax	Rmax	Rmax
→	Rmax	Rmax	Rmax	Rmax
↓	Rmax	Rmax	Rmax	Rmax
←	0	Rmax	Rmax	Rmax

	(1,1)	(1,2)	(2,1)	(2,2)
↑	0	0	0	0
→	0	0	0	0
↓	0	0	0	0
←	2	0	0	0

t=3: Agent moves up (s: (1,1))

Table 4: Known status, Reward counts, Transition counts

	(1,1)	(1,2)	(2,1)	(2,2)
↑	U	U	U	U
→	U	U	U	U
↓	U	U	U	U
←	K	U	U	U

	(1,1)	(1,2)	(2,1)	(2,2)
↑	Rmax [0]	Rmax	Rmax	Rmax
→	Rmax	Rmax	Rmax	Rmax
↓	Rmax	Rmax	Rmax	Rmax
←	0	Rmax	Rmax	Rmax

	(1,1)	(1,2)	(2,1)	(2,2)
↑	1	0	0	0
→	0	0	0	0
↓	0	0	0	0
←	2	0	0	0

t=4: Agent moves down (s: (2,1))

Table 5: Known status, Reward counts, Transition counts

	(1,1)	(1,2)	(2,1)	(2,2)
↑	U	U	U	U
→	U	U	U	U
↓	U	U	U	U
←	K	U	U	U

	(1,1)	(1,2)	(2,1)	(2,2)
↑	Rmax [0]	Rmax	Rmax	Rmax
→	Rmax	Rmax	Rmax	Rmax
↓	Rmax	Rmax	Rmax [0]	Rmax
←	0	Rmax	Rmax	Rmax

	(1,1)	(1,2)	(2,1)	(2,2)
↑	1	0	0	0
→	0	0	0	0
↓	0	0	1	0
←	2	0	0	0

t=5: Agent moves up (s: (1,1))

Table 6: Known status, Reward counts, Transition counts

	(1,1)	(1,2)	(2,1)	(2,2)
↑	K	U	U	U
→	U	U	U	U
↓	U	U	U	U
←	K	U	U	U

	(1,1)	(1,2)	(2,1)	(2,2)
↑	0	Rmax	Rmax	Rmax
→	Rmax	Rmax	Rmax	Rmax
↓	Rmax	Rmax	Rmax [0]	Rmax
←	0	Rmax	Rmax	Rmax

	(1,1)	(1,2)	(2,1)	(2,2)
↑	2	0	0	0
→	0	0	0	0
↓	0	0	1	0
←	2	0	0	0

t=6: Agent moves right (s: (2,1))

Table 7: Known status, Reward counts, Transition counts

	(1,1)	(1,2)	(2,1)	(2,2)
↑	K	U	U	U
→	U	U	U	U
↓	U	U	U	U
←	K	U	U	U

	(1,1)	(1,2)	(2,1)	(2,2)
↑	0	Rmax	Rmax	Rmax
→	Rmax	Rmax	Rmax [0]	Rmax
↓	Rmax	Rmax	Rmax [0]	Rmax
←	0	Rmax	Rmax	Rmax

	(1,1)	(1,2)	(2,1)	(2,2)
↑	2	0	0	0
→	0	0	1	0
↓	0	0	1	0
←	2	0	0	0

After $|S| \cdot |A|$ changes (this is a DDP):

Table 8: Known status, Reward counts, Transition counts

	(1,1)	(1,2)	(2,1)	(2,2)
↑	K	K	K	K
→	K	K	K	K
↓	K	K	K	K
←	K	K	K	K

	(1,1)	(1,2)	(2,1)	(2,2)
↑	0	0	0	1
→	0	0	0	1
↓	0	0	0	1
←	0	0	0	1

	(1,1)	(1,2)	(2,1)	(2,2)
↑	2	2	2	2
→	2	2	2	2
↓	2	2	2	2
←	2	2	2	2