CMPUT 652: Reinforcement Learning with Robots	**Fall 2019**
Lecture 7: September 24	
Instructor: Rupam Mahmood	*Scribe: Yufeng Yuan*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

7.1 Last Lecture

Previously, we understood reinforcement learning from the perspective of mind and from the first principle, we derived function approximation. The cases we encountered so far is the bandits, and a general formalization for reinforcement learning is introduced here.

7.2 Finite Markov Decision Process

Finite Markov Decision Process can be defined by a 5-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, p, \gamma \rangle$ and they are states, action space, reward, transition probabilities and discount rate respectively.

7.2.1 Agent-Environment Interaction

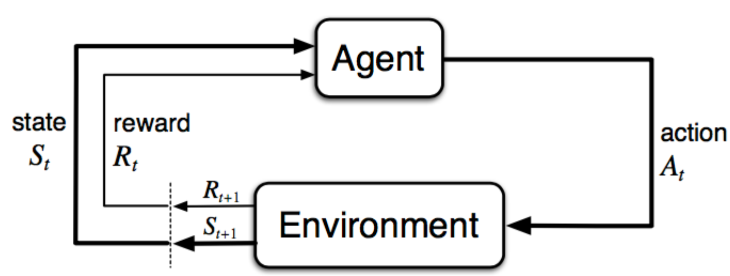


Figure 7.1: The agent-environment interaction in a Markov decision process

This interaction can be described as following: at each time step t, the agent receives some representation of the environment's state, $S_t \in \mathcal{S}$, and on that basis selects an action $A_t \in \mathcal{A}(s)$. One time step later, in part as a consequence of its action, the agent receives a numerical reward $R_{t+1} \in \mathcal{R}$, and finds itself in a new state S_{t+1}. Thus, the interaction of the agent and the environment will form a sequence or trajectory like: $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \cdots$.

7.2.2 Transition Probabilities and Return

The transition probabilities, also known as the dynamics of MDPs, are denoted by p:

$$p(s', r|s, a) = Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\}$$

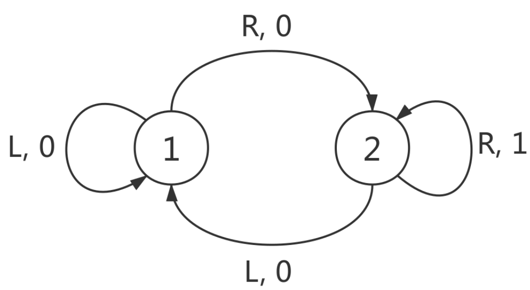
If p is known, one should be able to compute anything else about the environment because $p$completely characterizes the environment's dynamics. One thing should be noted here is that we assumed Markov Property here which future states is totally determined by current states.

At each time step, the goal of the agent is to maximize the cumulative future rewards which is defined as $G_t = R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$, where T is the final step. However, such formula doesn't apply to continuing tasks, where $T = \infty$, because G_t will be infinity too. To unify both episodic and continuing tasks, discount is introduced:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

where γ is a parameter $0 \le \gamma \le 1$ called the discount rate. The above can also be written in recursive form: $G_t = R_{t+1} + \gamma G_{t+1}$

7.2.3 A Small Example of MDP



$$\gamma = 0.9$$

$p(s', r\|s, a)$	1,0	1,1	2,0	2,1
$p(\cdot, \cdot\|1, L)$	1	0	0	0
$p(\cdot, \cdot\|1, R)$	0	0	1	0
$p(\cdot, \cdot\|2, L)$	1	0	0	0
$p(\cdot, \cdot\|2, R)$	0	0	0	1

7.3 Value Functions

Value functions, which tell the agent how good a state or a state-action pair is, are defined in terms of expected future rewards. The expectation in value functions is defined with respect to policy, which is a mapping from states to probabilities of selecting each possible action. Policies are denoted as $\pi(a|s)$ which is the probability that $A_t = a$ if $S_t = s$. The value-function of a state under policy π, denoted $v_\pi(s)$, is the expected return when starting in s and following π thereafter.

For MDPs, the value function is defined as:

$$v_\pi(s) = E_\pi[G_t|S_t = s] = E_\pi\Big[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}\Big|S_t = s\Big], s \in \mathcal{S}$$

The action-value function $q_\pi(s)$ is the expected return starting from s, taking the action a, and thereafter following policy π:

$$q_\pi(s) = E_\pi[G_t|S_t = s, A_t = a] = E_\pi\Big[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}\Big|S_t = s, A_t = a\Big], s \in \mathcal{S}, a \in \mathcal{A}(s)$$

7.3.1 Recursive Relationship of Value Functions

A fundamental property of value functions is that it can be written in a recursive form:

$$\begin{aligned}
v_\pi(s) &= E_\pi[G_t|S_t = s] \\
&= E_\pi[R_t + \gamma G_{t+1}|S_t = s] \\
&= \sum_a \pi(a|s)\sum_{s'}\sum_r p(s', r|s, a)\Big[r + \gamma E_\pi[G_{t+1}|S_{t+1} = s']\Big] \\
&= \sum_a \pi(a|s)\sum_{s'}\sum_r p(s', r|s, a)\Big[r + \gamma v_\pi(s')]\Big], s \in \mathcal{S}
\end{aligned}$$

7.4 Optimal Values and Policies

For finite MDPs, the optimal state-value function v_* is defined as:

$$v_*(s) = \max_\pi v_\pi(s)$$

the optimal action-value function q_* is defined as:

$$q_*(s) = \max_\pi q_\pi(s, a)$$

A policy π' is defined to be better than or equal to a policy π if its expected return is greater than or equal to that of π' for all states, which is $\pi' \geq \pi$ if and only if $v_{\pi'}(s) \geq v_\pi(s)$ for all $s \in \mathcal{S}$. There is always at least one policy that is better than or equal to all other policies, which is the optimal policy.

7.4.1 Greedification of Policies

For any arbitrary policy π, we can obtain an equal or better policy by greedification: $\pi'(s) = \arg\max_a q_\pi(s,a)$. If we use $g_\pi(s)$ and $u_\pi(s)$ to denote the value function with and without greedification on π:

$$\begin{aligned}
g_\pi(s) &= \max_a q_\pi(s,a) \\
&= q_\pi(s, \pi'(s)) \\
&= \sum_a \pi'(a|s) q_\pi(s,a) \\
u_\pi(s) &= \sum_a \pi(a|s) q_\pi(s,a) \\
g_\pi(s) &\geq u_\pi(s)
\end{aligned}$$

7.4.2 Bellman Optimality Function

The Bellman optimality function for optimal values functions is:

$$\begin{aligned}
v_*(s) &= \max_a E_{\pi_*}[G_t|S_t = s, A_t = a] \\
&= \max_a E_{\pi_*}[R_{t+1} + \gamma G_{t+1}|S_t = s, A_t = a] \\
&= \max_a E[R_{t+1} + \gamma v_*(S_{t+1})|S_t = s, A_t = a] \\
&= \max_a \sum_{s',r} p(s', r|s,a)[r + \gamma v_*(s')]
\end{aligned}$$

The Bellman optimality function for optimal action-values functions is:

$$\begin{aligned}
q_*(s,a) &= E\Big[R_{t+1} + \gamma q_*(S_{t+1}, a')\Big| S_t = s, A_t = a\Big] \\
&= \sum_{s',r} p(s', r|s,a)\Big[r + \gamma \max_{a}' q_*(s', a')\Big]
\end{aligned}$$

7.4.3 SGD to Estimate v_π

The objective function of mean squared error for estimating v_π is $E[(G_k - v)^2]$. The update rule for single sample SGD is $v_{k+1} = v_t - 2\alpha(G_k - v_k)$, where k is the time step so far from the beginning.