On Reinforcement Learning

Winston Wong

May, 2017

1 Reinforcement Learning Basic Model

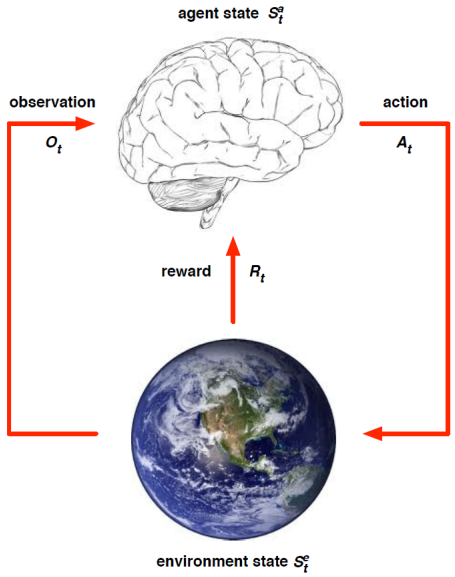

Figure 1: Basic Model of RL

The goal gere is to select a set of **actions** $\{A_1, A_2, A_3 ... A_t\}$ to maximize total future reward R_t based on the observations O_t. Since actions have long-term consequences and reward may be delayed, we cannot approach this problem greedily.

Definition **History** is the sequence of observation, actions and rewards. i.e.

$$H_t = A_1, O_1, R_1, A_2, O_2, R_2, ..., A_t, O_t, R_t$$

Definition A **state** contains the information that determines what happens next (i.e. $A_{t+1}, O_{t+1}, R_{t+1}$), and is defined as a function of the history.

$$S_t = f(H_t)$$

Definition The **environmental state**, S_t^e , is the information that determines the next observation O_{t+1} and reward R_{t+1} produce by the environment. Usually not visible to agent.

Defintition The **agent state**, S_t^a, is the information that determines the next action A_{t+1} produce by the agent, which is often a function of history. i.e. $S_t^a = f(H_t)$

1.1 Information State

Definition The **information state** (a.k.a **Markov state**) contains all useful information from the history. Given the present, the future is independent of the past. The Markove state is a sufficient statistic of the future. More formally, the state S_t is **Markov** i.f.f

$$\mathbb{P}[S_{t+1}|S_t] = \mathbb{P}[S_{t+1}|S_1, S_2, S_3, ..., S_{t-1}, S_t] \tag{1}$$

The environmental state is Markov (by definition).

Proposition The history is also Markov.

Proof

$$\begin{aligned}\mathbb{P}[H_{t+1}|H_1, ..., H_t] =& \mathbb{P}[H_{t+1}|A_1, O_1, R_1, A_1, O_1, R_1, A_2, O_2, R_2, ..., \\ & A_1, O_1, R_1, A_2, O_2, R_2, ..., A_t, O_t, R_t] \\ =& \mathbb{P}[H_{t+1}|A_1, O_1, R_1, A_2, O_2, R_2, A_3, O_3, R_3..., A_t, O_t, R_t] \\ =& \mathbb{P}[H_{t+1}|H_t]\end{aligned}$$

1.2 Observability

An environment is **fully observable** if agent can directly observe the environment state. i.e. $S_t^a = O_t = S_t^e$. Formally known as Markov Decision Process (MDP).
An environment is **partially observable** agent indirectly observe the environment. $S_t^a \neq S_t^e$. Formally known as Partially Observable Markov Decision Process (POMDP). We can construct the agent state S_t^a by using:

- Complete History: $S_t^a = H_t$
- Beliefs on environment state: $S_t^a = (\mathbb{P}[S_t^e = s_1], ..., \mathbb{P}[S_t^e = s_n])$
- Recurrent Neural Network $S_t^a = \sigma(S_{t-1}^a W_s + O_t W_o)$

1.3 Components of RL agent

Definition Policy is the mapping of **state** to **action**

- Deterministic Policy: $a = \pi(s)$
- Probabilisitc Policy $\pi(\mathbf{a}|\mathbf{s}) = \mathbb{P}[A_t|S_t]$

Definition A **Value Function** is a function **associate with each policy** used to evalulate how good the states are, therefore use to select actions. Formally,

$$v_\pi(s) = \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ...|S_t = s]$$

Definition A model predicts what the **environment** will do next.

- Transition Probability. The probaility of going from a state to another.

$$\mathcal{P}^a_{ss'} = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$$

- Emission Probability. The expected reward obtained from the current state.

$$\mathcal{R}^a_s = \mathbb{E}[R_t | S_t = s, A_t = a]$$

1.4 Learning and Planning

If the environment is initially unknown, then we need **learning** by interact with the environment to improve the policy. A.k.a Exploration.
Conversely, if the environment is known, all wee need is to perform computations (query) the model to improve the policy. This is known as **planning** (a.k.a exploitation).

1. Query the emulator.
2. If action a is taken from state s, what is the score and what is the next state?
3. Do tree search to find the optimal policy.

2 Markov Decision Process

2.1 Introduction to MDP

Recall that a Markvo Decision Process (MDP) is characterized by its fully observable environment. That being said, any RL problems can be transformed into MDP.
The Markov Decision Process inherits from Markov Reward Process (MRP), which extends from the Markov Process (Markov Chain).

Definition Markov Process (MP) is the tuple $\langle \mathcal{S}, \mathcal{P} \rangle$ where

- $\mathcal{S}$ is the finite set of states
- $\mathcal{P}$ is the state transitional probability matrix. where an entry $\mathcal{P}^a_{ss'}$ is defined
$$\mathcal{P}_{ss'} := \mathbb{P}[S_{t+1} = s' | S_t = s]$$

Definition Markov Reward Process (MRP) is the tuple $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ where

- $\mathcal{R}$ is the reward function
$$\mathcal{R}_s = \mathbb{E}[R_t | S_t = s]$$
- γ is the discount factor, $\gamma \in [0, 1]$

Definition The return G_t (not to be confused with reward R_t) is the total discounted reward from step t

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + ... = \sum_{i=0}^{\infty} \gamma^i R_{t+i}$$

Definition The state value function $v(s)$ of an MRP gives the expected return.

$$\begin{aligned} v(s) &= \mathbb{E}[G_t | S_t = s] \\ &= \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... | S_t = s] \\ &= \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) | S_t = s] \\ &= \mathcal{R}_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'} v(s') \end{aligned}$$

Notice the equation for the value function casted in matrix form is known as **Bellman Equation**. Concretely,

$$\begin{aligned} v &= \mathcal{R} + \gamma \mathcal{P} v \\ (\mathrm{I} - \gamma \mathcal{P}) v &= \mathcal{R} \\ v &= (\mathrm{I} - \gamma \mathcal{P})^{-1} \mathcal{R} \end{aligned}$$

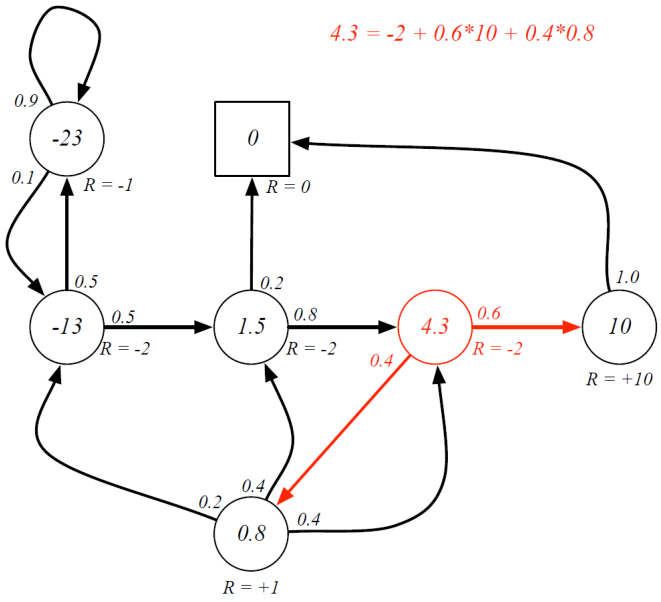

Figure 2: Value function of a Markov Reward Process

2.2 Markov Decision Process

Finally, let's dig into the topic of today - the Markov Decision Process (MDP). MDP is an MRP with decisions, and states are Markov.

Definition The Markov Decision Process (MDP) is the tuple $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma, \mathcal{A} \rangle$ where

- $\mathcal{S}$ is the finite set of states
- $\mathcal{A}$ is the set of finite actions
- $\mathcal{P}$ is the state transitional probability matrix. where an entry $\mathcal{P}^a_{ss'}$ is defined
$$\mathcal{P}^a_{ss'} := \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$$
- $\mathcal{R}$ is the reward function
$$\mathcal{R}^a_s = \mathbb{E}[R_t | S_t = s, A_t = a]$$
- γ is the discount factor, $\gamma \in [0, 1]$

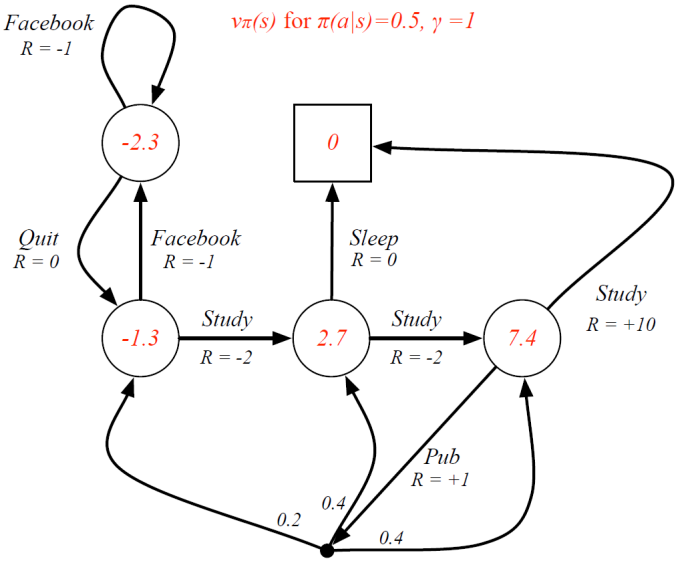

Figure 3: A Markov Decision Process

Definition A policy (probabilistic) given by

$$\pi(\mathbf{a}|\mathbf{s}) = \mathbb{P}[A_t|S_t]$$

defines the behavior of the agent, is stationary and dependent of the current state (not history).

Note that the MDP $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma, \mathcal{A} \rangle$ can be reduced to the MRP $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ where

- $\mathcal{P}^{\pi}_{ss'} = \sum_{a \in \mathcal{A}} \pi((a|s)) \mathcal{P}^{a}_{ss'}$
- $\mathcal{R}^{\pi}_{s} = \sum_{a \in \mathcal{A}} \pi((a|s)) \mathcal{R}^{a}_{ss'}$

Definition The *state-value* function is the expected return from state s following policy π

$$v_{\pi}(s) = \mathbb{E}[G_t|S_t = s]$$

Definition The *action-value* function is the expected return from state s, taking action a, and following policy π

$$q_{\pi}(s, a) = \mathbb{E}[G_t|S_t = s, A_t = a]$$

The inter-dependent relationship of The action-value and state-value func-

tions are captured by the following equation.

$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(\mathbf{a}|\mathbf{s}) q_\pi(s, a)$$
$$q_\pi(s, a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{ss'}^a v(s')$$

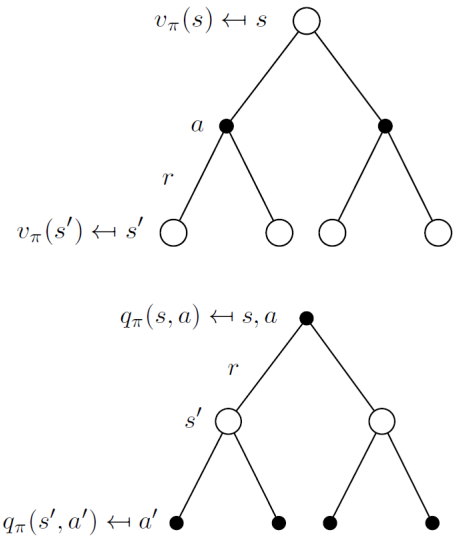

Figure 4: Recursive relation of value functions

Expressing the equations above in a recursive manner, we have the **Bellman Expectation Equation**

$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(\mathbf{a}|\mathbf{s})(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{ss'}^a v(s'))$$
$$q_\pi(s, a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{ss'}^a (\sum_{a' \in \mathcal{A}} \pi(\mathbf{a}'|\mathbf{s}') q_\pi(s', a'))$$

Definition The **state-value function** $v_*(s)$ is said to be **optimal** if

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

Similarly, The **action-value function** $q_*(s, a)$ is said to be **optimal** if

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

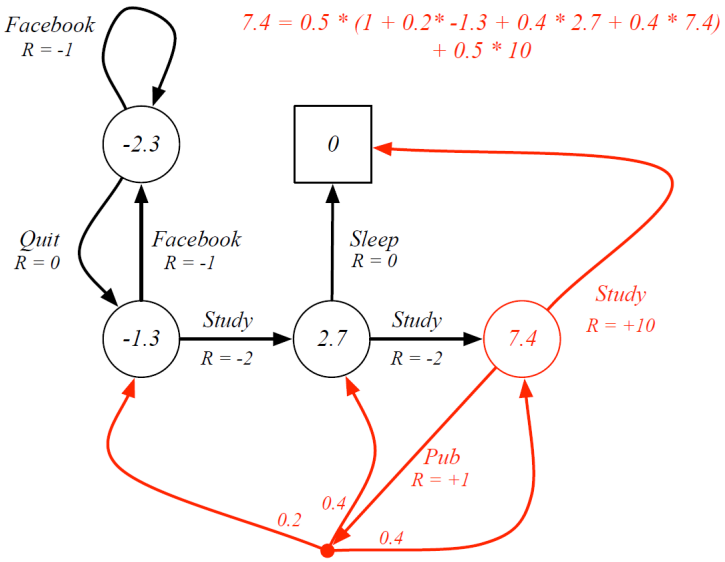

Figure 5: Computing state-value function using the Bellman Expectation Equation

Theorem Define $\pi \geq \pi'$ iff $v_\pi(s) \geq v_\pi|(s)$, for any Markov Process,

- $\exists\, \pi_*$ such that $\pi_* \geq \pi$
- All optimal policies achieve the same optimal value function $v_{\pi_*}(s) = v_*(s)$
- All optimal policies achieve the same optimal action-value function $q_{\pi_*}(s,a) = q_*(s,a)$

Theorem An optimal policy can be found by maximizing over $q_*(s,a)$

$$\pi_*(\mathbf{a}|\mathbf{s}) = \begin{cases} 1, if a = \text{argmax}_{a \in A} q_*(s,a) \\ 0, otherwise \end{cases}$$

Finally, we summarize this chapter by introducing the **Bellman optimality Equation**

Definition The recursive definition of the optimal value function is given as the following:

$$v_*(s) = \max_a q_*(s,a) = \max_a (\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s'))$$

$$q*(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \max_{a'} q_*(s', a')$$

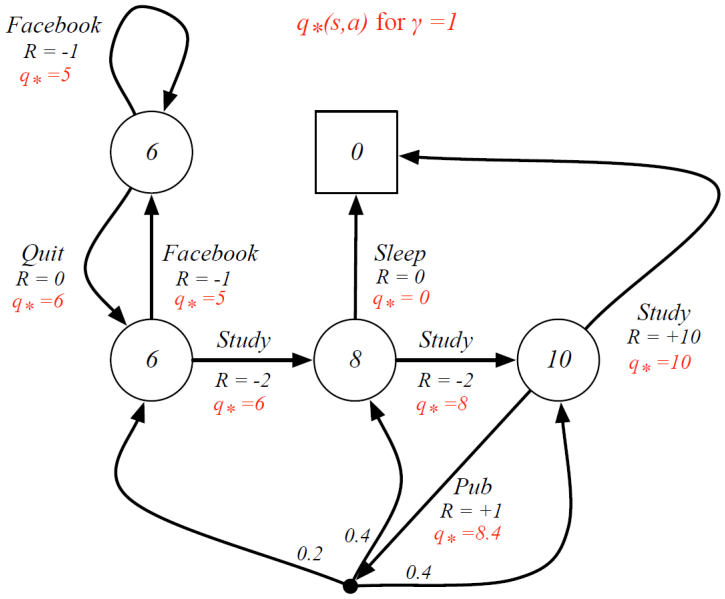

Figure 6: Computing optimal action-value function using the Bellman Optimality Equation

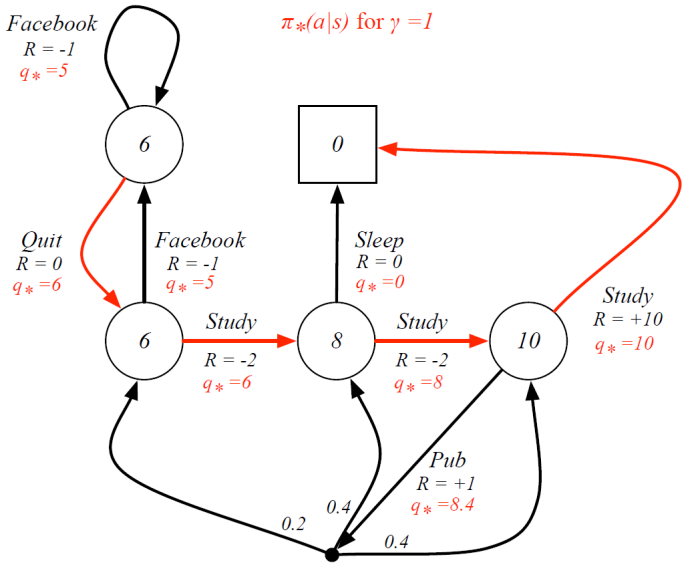

Figure 7: Finding the Optimal Policy

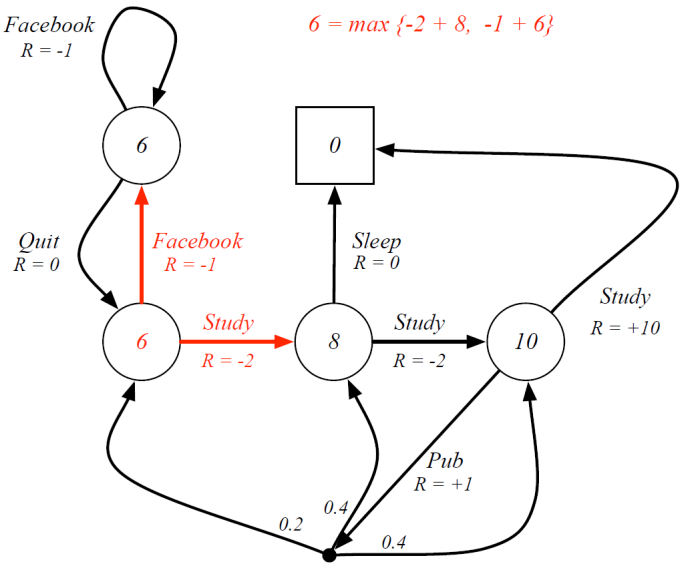

Figure 8: Finding Optimal value function $v_*(s)$