Summary Chapter 3 & 4 RL Book [?] Submitted by: Dhawal Gupta at September 29, 2019

1 Markov Decision Process (MDP)

MDP's are a mathematically idealized form of RL problems, finite MDP constitute of a finite set of States ($\mathcal{S}$), finite set of Actions ($\mathcal{A}$) and finite set of possible rewards ($\mathcal{R}$), also $s' \in \mathcal{S}^+$ is used to represent the set of states plus the terminal state , and every element in those sets have a well defined discrete probability.

A sample trajectory looks like $S_0, A_0, R_1, S_1, A_1, R_2, S_2, \ldots$ where an action $A_t \in \mathcal{A}$ in State $S_t \in \mathcal{S}$ in the environement returns a next state $S_{t+1} \in \mathcal{S}$ and a reward $R_{t+1} \in \mathcal{R}$.

The **dynamics** of the MDP are defined as

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

which is a deterministic discrete probability distribution. Where each next state and reward are completely dependent only on the preceding state, and no state before that (restriction of S to be expressive). And we call the state to have **Markov Property**.

We can express some other functions using **??**

$$p(s'|s, a) \doteq \sum_{r \in \mathcal{R}} p(s', r|s, a) \tag{2}$$

$$r(s, a) \doteq \mathbb{E}[R_t|s, a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r|s, a) \tag{3}$$

$$r(s, a, s') \doteq \mathbb{E}[r_t|s, a, s'] = \sum_{r \in \mathcal{R}} r \frac{p(s', r|s, a}{p(s'|s, a)} \tag{4}$$

]The general rule being anything that cannot be changed arbitrarily by the agent is considered to be outside of it and part of the environment, like robot limbs, reward signals etc. The agent-environment boundary represents the limit of the agents absolute control not of its knowledge.

Some important points I think . we should know

1. Reward signal is not the place to impart to the agent prior knowledge about how to achieve what we want to do

1.1 Return and Episode

The return G_t that the agent tries to maximize is often defined as

$$G_t \doteq R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots \tag{5}$$

$$G_t = R_t + \gamma G_{t+1} \tag{6}$$

$$= \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{7}$$

$$= \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k \text{ for episodic cases} \tag{8}$$

where gamma is defined as the discounting factor to avoid running into infinities.

1.2 Unifying notation for episodic and continuing tasks

WE can use the same notation for episodic tasks by putting the constraint that at the ending time T, the agent enters an absorbing state with zero reward and we condition following as $T = \infty$ or $\gamma = 1$ but not both.

1.3 Policies and Value Functions

Value Function : of state s under policy π denoted by $\mathrm{v}_\pi(s)$ defined as is the expected return starting in that state and following π from there.

$$\mathrm{v}_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s] \forall s \in \mathcal{S} \tag{9}$$

Important : The value of the terminal state, if any, is always zero. Whereas $\mathrm{q}_\pi(s, a)$ is defined the expected run of taking action a in s and then following policy π.

$$\mathrm{q}_\pi(s, a) \doteq \mathbb{E}_\pi[G_t|S_t = s, A_t = a] \tag{10}$$

1.4 Recursive form of $\mathrm{v}_\pi(s)$ and $\mathrm{q}_\pi(s, a)$

Bellman equation for value function under policy π. The value function v_π is the unique solution to its Bellman equation.

$$\mathrm{v}_\pi(s) = \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a)[r + \gamma\, \mathrm{v}_\pi(s')] \forall s \in \mathcal{S} \tag{11}$$

$$= \mathbb{E}_\pi[q(s, A_t)] \tag{12}$$

$$\mathrm{q}_\pi(s, a) = \sum_{s', r} p(s', r|s, a)[r + \gamma \sum_{a}^{'} \pi(a'|s')\, \mathrm{q}_\pi(s', a')] \forall s \in \mathcal{S} \tag{13}$$

$$= \mathbb{E}_\pi[R(s, a) + \gamma\, \mathrm{v}_\pi(S_{t+1})|S_t = s, A_t = a] \tag{14}$$

1.5 Optimal Policies and Value Functions

A better or equal policy π over π' is defined as $\pi \geq \pi'$ iff $\mathrm{v}_\pi(s) \geq v_{\pi'}(s) \forall s \in \mathcal{S}$. There is always one policy better than a policy and an optimal policy π^* which is better than any other policy(can be more than one). and share the same state value function. i.e. $v_*(s) \doteq \underset{\pi}{max}\, \mathrm{v}_\pi(s)$ and $q_*(s, a) \doteq \underset{\pi}{max}\, \mathrm{q}_\pi(s, a)$ and $q_*(s, a). = \mathbb{E}[R_{t+1}. + \gamma v_*(S_{t+1})|S_t = s, A_t = a]$ The value of a state under an optimal policy must equal the expected return for best action from that state.

$$v_*(s) = \underset{a \in \mathcal{A}}{max} q_{\pi*}(s, a)$$

$$= \underset{a}{max} \sum_{s', r} p(s', r|s, a)[r + \gamma . v_*(s')] \tag{15}$$

Similarly for the q value function we get.

$$q_*(s, a) = \mathbb{E}[R_{t+1} + \gamma \underset{a'}{max} q_*(S_{t+1}, a')|S_t = s, A_t = a]$$

$$= \sum_{s', r} p(s', r|s, a)[r + \gamma \max_{a'} q_*(s', a')] \tag{16}$$

A policy that is greedy with respect to v_* will be an optimal policy. Having the knowledge of $q_*(s, a)$ eliminates the need to do a one step search on the actions, and value of future states, and also the environment dynamics.

2 Dynamic Programming

DP can be used to compute optimal policies given a perfect model of the environment as an MDP , but their highly compute intensive.

2.1 Policy Evaluation

The value function for a policy is know as policy evaluation and normally equation **??** can be thought of as a system of $|S|$ equations, where each term is a linear dependent on the other variables, and there are $|S|$ unknowns. There are also two methods to solve this, 1. Iterative 2. Analytical, we describe the iterative method. We choose initial approaximation for each state v_0 (except for terminal state, which is 0) and use the bellman equation for the update rule.

$$v_{k+1}(s) \doteq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}|S_t = s] \tag{17}$$

$$= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_k(s')] \tag{18}$$

All the updates done on DP algorithms are called expected updates because they are based on expectation of the next states rather than sampling of the next states.

2.2 Policy Improvement

We compute a value function as to improve the existing policy. Determining v_π for a policy π, we want to know wether we can improve on the policy i.e. take action $a \neq \pi(s)$ and improve our value function in general. One way of seeing this is taking action a in s and then following policy π afterwards, which can be written as : $\mathrm{q}_\pi(s, a)$ where a might not be sampled from π. Criterion being if this greater than $\mathrm{v}_\pi(s)$ and then we can replace this step with always taking action as it will always imprive the value function for $v_{\pi'}(s)$, where the new policy can be taking action $a = \pi'(s)$ and then following π for rest.

Policy Improvement Theorem π and π' are 2 deterministic policies, such that $\forall s \in \mathcal{S}$

$$\mathrm{q}_\pi(s, \pi'(s)) \geq \mathrm{v}_\pi(s) \tag{19}$$

Then the policy π' must be as good as, or better than π. This is a generalisation of the above statement.

$$v_{\pi'}(s) \geq \mathrm{v}_\pi(s) \tag{20}$$

Proof the same has been expanded in great detail later. Now we can extend the single change in action to all states, considering a new greedy policy π's given by,

$$\pi'(s) \doteq \underset{a}{\text{argmax}}\ \mathrm{q}_\pi(s, a) \tag{21}$$

, where we can distribute the policy probability equally on the multiple best action it is okay until and unless all the suboptimal actions get a 0 probability value. This meets with the condition of the Policy Improvement Theorem **??**. This process of greedyfying is often called as policy improvement.

Getting the Optimal Policy : Suppose new policy π' is as good as and not better than π i.e. $\mathrm{v}_\pi = \mathrm{v}_{\pi'}$ and from **??** we get $\forall s \in \mathcal{S}$.

$$\mathrm{v}_{\pi'}(s) = \max_{\mathrm{a}} \sum_{s',r} p(s', r|s, a)[r + \gamma\, \mathrm{v}_{\pi'}(s')]$$

Which matches the Bellman Optimality equation there $\mathrm{v}_{\pi'} = v_*$

2.3 Policy Iteration

The process of alternating between policy evaluation and policy improvement to approach the optimal policy, is called as policy iteration. A policy π can be improved using v_π to get a new policy π', as summarized in this diagram

$\pi_0 \xrightarrow{E} v_{\pi_0}$

3 Proofs

3.1 Proof of $\mathrm{v}_\pi(s)$

$$\begin{aligned}
\mathrm{v}_\pi(s) &= \mathbb{E}_\pi[G_t|S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1}|S_t = s] + \gamma\, \mathbb{E}_\pi[G_{t+1}|S_t = s] \\
&= \sum_{r,a} r \times \pi(a|s) \times p(r|s, a) + \gamma\, \mathbb{E}_\pi[G_{t+1}|S_t = s] \\
&= \sum_{r,a} r\pi(a|s)p(r|s, a)+ \\
&\gamma(\sum_{s'}(\sum_a \pi(a|s)p(s'|s, a)\, \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s'])) \\
&= \sum_a \pi(a|s)[\sum_r rp(r|s, a)+ \\
&\gamma \sum_{s'} p(s'|s, a)\, \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']] \\
&= \sum_a \pi(a|s)[\sum_{r,s'} rp(s', r|s, a)+ \\
&\gamma \sum_{s',r} p(s', r|s, a)\, \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']] \\
&= \sum_a \pi(a|s) \sum_{r,s'} p(s', r|s, a)[r + \gamma\, \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma\, \mathrm{v}_\pi(s')]\forall s \in \mathcal{S}
\end{aligned}$$

3.2 Proof of policy Improvement (I have done this in the exercise solutions)

$$\begin{aligned}
\mathrm{v}_\pi(s) &\leq \mathrm{q}_\pi(s, \pi'(s)) \\
&= \mathbb{E}_{R_{t+1},S_{t+1}\sim p(s',r|s,a)}[R_{t+1} + \gamma\, \mathrm{v}_\pi(S_{t+1})|S_t = s, A_t = \pi'(s \\
&= \mathbb{E}_{R_{t+1},S_{t+1}\sim p(s',r|s,a),a\sim\pi'}[R_{t+1} + \gamma\, \mathrm{v}_\pi(S_{t+1})|S_t = s] \\
&\text{Shortening the notation} \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma\, \mathrm{v}_\pi(S_{t+1})|S_t = s] \\
&\text{Applying } \pi' \text{ on another step we get} \\
&\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma\, \mathrm{q}_\pi(S_{t+1}, \pi'(S_{t+1})|S_t = s] \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma\, \mathbb{E}_{\pi'}[R_{t+2} + \gamma\, \mathrm{v}_\pi(S_{t+2})|S_{t+1}]|S_t = s] \\
&\text{Continued in the exercise solutions}
\end{aligned}$$

3.3 Notation

We can define different aspects of the problem as follows:

a	The value of action selected
A_t	Random variable, action selected at time t
R_t	Reward collected at time t
$Q_t(a)$	Estimate value function of action a at time t
$q_*(a)$	Optimal value of action a
α	Constant learning rate
α_t	Learning rate at time t
$\doteq$	Defined as t
π_t	Policy followed at t

The optimal value function i.e. q_* for an action a can be defined as follows

$$q_*(a) \doteq \mathbb{E}[R_t|A_t = a] \tag{22}$$

3.4 Terms

Exploitation Policy : A policy π_t which tries to exploit the estimated value of an action at time t using Q_t and picks the action with the estimated highest return.
Exploration Policy : A policy π_t which tries to explore the different actions at time t and picks action using some criterion or randomly.
Stationary Problem : A k-armed bandit problem where the reward distribution of each arm stays unchanged over the duration of experiment.
Non Stationary Problem : A k-armed bandit problem in which the reward distribution of each arm (/action) may change over the period of time.

4 Action-value Methods

We can use sample averaging method to estimating the value of different actions

$$Q_t(a) \doteq \frac{\sum_{i=1}^{t-1} R_i \nVdash_{A_i=a}}{\sum_{i=1}^{t-1} \nVdash_{A_i=a}} \tag{23}$$

Either we can follow a greedy policy and select our action based on $argmax_a Q_t(a)$, or we can follow an ϵ-greedy policy, where we select a random action with probability ϵ otherwise we pick do exploitation

5 Incremental Implementation

Rather than keeping a track of all the previous rewards for a action a we will write it into a form of incremental form where we need to remember the previous estimate and number of times the action as been picked.

$$Q_{n+1} = Q_n + \frac{1}{n}[R_n - Q_n] \text{ for a single action} \tag{24}$$

6 Tracking a Non Stationary Problem

The problem with **??** is that the update error weight decays with the number of plays and the agent might not be able to adapt to changing reward distributions after a given amount of time. We can try to use a constant learning parameter α to counter this.

$$\begin{aligned} Q_{n+1} &\doteq Q_n + \alpha[R_n - Q_n] \\ &= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i. \end{aligned} \tag{25}$$

7 Optimistic Initial Values

The book [**?**] shows that setting the initial action values estimates of all the actions to an optimistic values allows us to embed the process of exploration in the inherent process of doing a greedy search , but again that is plagued by the process of non stationary problems.

8 Upper-Confidence-Bound Action Selection

In this approach we try to reason on the uncertainty of different action and their potential of returning higher rewards than the current estimated maximum reward. One way to do it is via following:

$$A_t \doteq \underset{a}{argmax}[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}}] \tag{26}$$

Where $N_t(a)$ denotes the number of times action a has been selected prior to time t. c control the degree of exploration.

9 Gradient Bandit Algorithm

Now we try to learn a numerical preference for each action a denoted by $H_t(a)$. Larger preference implies more probability of taking the action, there is not interpretation in terms of reward. Only the relative preference matters and absolute values carry no significance.

$$PrA_t = a \doteq \frac{\exp^{H_t(a)}}{\sum_{b=1}^{k} exp^{H_t(b)}} \doteq \pi_t(a) \tag{27}$$

Update rules for the preference of action.

$$\begin{aligned} H_{t+1}(A_t) &\doteq H_t(A_t) + \alpha(R_t - \hat{R}_t)(1 - \pi_t(A_t)), \text{and} \\ H_{t+1}(a) &\doteq H_t(a) - \alpha(R_t - \hat{R}_t)\pi_t(a), \text{ for all } a \neq A_t \end{aligned} \tag{28}$$

Here $\hat{R}_t$ is the average of all rewards up to the time t, including it.