Dynamic Programming methods are suitable for MDPs for which the model dynamics are known. In this chapter, we will discuss the methods that can be used to solve MDPs for which the model dynamics are not known. These methods are called Monte Carlo methods. The main idea behind Monte Carlo methods is to estimate the value function by averaging the returns obtained from the episodes. The main advantage of Monte Carlo methods is that they do not require the model dynamics to be known. The main disadvantage of Monte Carlo methods is that they are computationally expensive.

0.1 Monte Carlo Prediction

In earlier chapters, we defined value function as expected sum of discounted rewards. To estimate the expected sum, one way is to average the discounted rewards obtained from the episodes.This is the idea behind the monte carlo methods. Even the core idea of averaging remains same, there are simple variations in the way we consider the rewards for averaging.
Let us discuss two simple algorithms.

- First-visit MC

- Every visit MC

Algorithm 1 First-visit MC

Initialize $V(s)$ arbitrarily for each $s \in S$, and policy π to be evaluated
Initialize $Returns(s), N(S_t)$ as empty list, for each $s \in S$
while each episode **do**:
    Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \cdots S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    while Loop for each step of episode $t = T-1, \cdots, 2, 1, 0$ **do**:
        $G \leftarrow \gamma G + R_{t+1}$
        Unless S_t is already in $S_0, S_1, \cdots, S_{t-1}$:
            Append G to $Returns(S_t)$
            $N(S_t) \leftarrow N(S_t) + 1$
            $V_{n+1}(S_t) \leftarrow V_n(S_t) + \frac{1}{N(S_t)} [G - V_n(S_t)]$
    end while
end while

First visit MC averages the rewards following from the first visit of the state.

Algorithm 2 Every-visit MC

Initialize $V(s)$ arbitrarily for each $s \in S$, and policy π to be evaluated
Initialize $Returns(s), N(S_t)$ as empty list, for each $s \in S$
while each episode **do**:
    Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \cdots S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    while Loop for each step of episode $t = T-1, \cdots, 2, 1, 0$ **do**:
        $G \leftarrow \gamma G + R_{t+1}$
        Append G to $Returns(S_t)$
        $N(S_t) \leftarrow N(S_t) + 1$
        $V_{n+1}(S_t) \leftarrow V_n(S_t) + \frac{1}{N(S_t)} [G - V_n(S_t)]$
    end while
end while

Every visit MC averages the rewards following every visit of the state.

One can modify the algorithm to find the state action value function. The algorithm is same as above except that we have to maintain the state action value function instead of state value function.

Monte Carlo methods doesn't bootstrap and the estimated value function is state independent. It means that monte carlo method doesn't use the estimate of other states to estimate the value of a state. It is also called on-policy method because it uses the same policy to generate the episodes and to evaluate the policy.

0.2 Monte Carlo Control

We now have the algorithm to evaluate any given policy π in unknown dynamics MDP. We can use those algorithm to evaluate the policy π and then improve the policy. This is the idea behind Monte Carlo control.Using the same strategy used in policy improvement algorithm, we find the values function for a given policy and then act greedy with respect to the value function which improves the earlier policy. We repeat this process until we find the optimal policy.

There are inherent assumptions in the above algorithms. One assumption is exploring starts, all the state-action pairs are explored enough to average the rewards. Another assumption is we have to sample infinite episodes to converge. The secon assumption we can relax by following the general policy improvement algorithm.

We will see modified version of first visit MC algorithm to find the optimal policy. The algorithm is same as first visit MC algorithm except that we use the greedy policy with respect to the value function to generate the episodes.

Algorithm 3 First Visit MC with policy improvement

Initialize $Q(s,a)$ arbitrarily for each $s \in S, a \in A$, and policy π to be evaluated
Initialize $Returns(s,a), N(s,a)$ as empty list, for each $s \in S, a \in A$
while each episode **do**:
    Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \cdots S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    while Loop for each step of episode $t = T-1, \cdots, 2, 1, 0$ **do**:
        $G \leftarrow \gamma G + R_{t+1}$
        Unless S_t is already in $S_0, S_1, \cdots, S_{t-1}$:
            Append G to $Returns(S_t)$
            $N(S_t) \leftarrow N(S_t) + 1$
            $Q_{n+1}(S_t, A_t) \leftarrow Q_n(S_t, A_t) + \frac{1}{N(S_t)} [G - Q_n(S_t, A_t)]$
            $\pi(S_t) \leftarrow argmax_a Q_{n+1}(S_t, A_t)$
    end while
end while

Now, we will apply the same policy improvement idea on every visit MC algorithm.

Algorithm 4 Every Visit MC with policy improvement

Initialize $Q(s,a)$ arbitrarily for each $s \in S, a \in A$, and policy π to be evaluated
Initialize $Returns(s,a), N(s,a)$ as empty list, for each $s \in S, a \in A$
while each episode **do**:
    Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \cdots S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    while Loop for each step of episode $t = T-1, \cdots, 2, 1, 0$ **do**:
        $G \leftarrow \gamma G + R_{t+1}$
        Append G to $Returns(S_t, A_t)$
        $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$
        $Q_{n+1}(S_t, A_t) \leftarrow Q_n(S_t, A_t) + \frac{1}{N(S_t)} [G - Q_n(S_t, A_t)]$
        $\pi(S_t) \leftarrow argmax_a Q_{n+1}(S_t, A_t)$
    end while
end while

From sutton and barto, the convergence to an optimal policy is inevitable. However, there is no formal proof to show the convergence.

To relax the first assumption, we should make sure that policy is epsilon soft. $\pi(\cdot|S_t) > 0$. This is acheived by using ϵ-greedy policy.In contrast to the greedy approach, we do not choose action greedily rather we choose action with probability ϵ randomly and with probability $1 - \epsilon$ greedily. This is called ϵ-greedy policy. We will see the algorithm to find the optimal policy using ϵ-greedy policy.

Algorithm 5 Every Visit MC with ϵ-greedy policy improvement

Initialize $Q(s,a)$ arbitrarily for each $s \in S, a \in A$, and policy π to be evaluated
Initialize $Returns(s,a), N(s,a)$ as empty list, for each $s \in S, a \in A$
while each episode **do**:
    Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \cdots S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    while Loop for each step of episode $t = T-1, \cdots, 2, 1, 0$ **do**:
        $G \leftarrow \gamma G + R_{t+1}$
        Append G to $Returns(S_t, A_t)$
        $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$
        $Q_{n+1}(S_t, A_t) \leftarrow Q_n(S_t, A_t) + \frac{1}{N(S_t)}\left[G - Q_n(S_t, A_t)\right]$
        $\pi(a|S_t) \leftarrow \begin{cases} 1 - \epsilon + \frac{\epsilon}{|A|} & \text{if } a = argmax_a Q_{n+1}(S_t, A_t) \\ \frac{\epsilon}{|A|} & \text{otherwise} \end{cases}$
    end while
end while

From sutton and barto, the above epsilon-greedy policy improvement algorithm also converges to an optimal policy. Till now, we have seen the on-policy algorithms to find the optimal policy. On-policy algorithms suffer from disadvantage that it can't reuse the generated episodes. To overcome this disadvantage. On-policy approach means we use a policy to generate episodes and use the same policy to improve the policy. Off-policy approach means we use a policy called *behaviour policy* to generate episodes and use another policy called *target policy* to improve continously.

we first solve the prediction problem of value function using off-policy approach. We will use the same algorithm as we used for on-policy approach. The only difference is that we will use the behaviour policy to generate episodes and use the target policy to improve the policy. we make an assumption that $b(a|s) > 0$ to facilitate the exploring starts behaviour. On the otherhand, target policy can be deterministic. While the behaviour policy must be stochastic to facilitate the exploration.

Importance Sampling is the trick followed by all off-policy algorithms. With the help of importance sampling, we can reuse the generated episodes. Instead of accumulting the discounted sum of rewards, we will accumulate the weighted discounted sum of rewards. The weight is the ratio of the probability of the action taken by the behaviour policy to the probability of the action taken by the target policy.

Given a starting state S_0 under a stochastic policy π, the resulting trajectory will have a probability distribution

$$P_\pi(S_0, A_0, R_1, S_1, A_1, R_2, \cdots, S_{T-1}, A_{T-1}, R_T) = \prod_{t=0}^{T-1} \pi(A_t|S_t) \cdot P(S_{t+1}|S_t, A_t) \cdot R_{t+1}$$

We define the relative probability of the trajectory under the target policy π and the behaviour policy b as

$$\begin{aligned} \rho_{t:T-1} &= \frac{\prod_{k=t}^{T-1} \pi(A_k|S_k) Pr(S_{k+1}|S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k|S_k) Pr(S_{k+1}|S_k, A_k)} \\ &= \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)} \end{aligned}$$

This is called importance sampling ratio.If you sample with behaviour policy b and then estimate $V(s)$ as $\mathbb{E}[G_t|S_t]$ which will result in $V_b(s)$. Nevertheless, we need to estimate $V_\pi(s)$ which is the value function under the target policy. We can use the importance sampling ratio to estimate $V_\pi(s)$ as expected weighted sum of rewards, where the weight is given by the importance sampling ratio. Instead of $\mathbb{E}[G_t|S_t]$, you approximate $\mathbb{E}[\rho_{t:T-1}G_t|S_t] = V_\pi(s)$. ρ is the importance sampling ratio.

Off-policy Every Visit MC Prediction Algorithm needs to track where the state-action pair occurs in the episode. Let it be $T(s,a)$. We store the time steps at which the state-action pair occurs in the episode. This favours us to calculate the importance sampling ratio from those time steps. We will use the importance sampling ratio to calculate the weighted sum of rewards. This will give us the value function under the target policy.

$$Q_\pi(s,a) = \frac{\sum_{t\in T(s,a)} \rho_{t:T(t)-1} G_t}{|T(s,a)|}$$

This is called ordinary importance sampling.

$$Q_\pi(s,a) = \frac{\sum_{t\in T(s,a)} \rho_{t:T(t)-1} G_t}{\sum_{t\in T(s,a)} \rho_{t:T(t)-1}}$$

This is an alternative way of calculating the value function under the target policy. This is called weighted importance sampling. Now the question is which algorithm to follow or which algorithm leads to convergence and at what rate.

Let's see in case of first visit algorithms. The differences are generally expressed in terms of biases and variances. Ordinary importance sampling is unbiased wheras weighted importance sampling is biased(although the bias converges to zero finally). Ordinary importance sampling has unbounded variance. On the other hand, weighted importance sampling had lower variance and is the preferred algorithm. Nevertheless, we will use the ordinary importance sampling while using approximate value function methods.

In every visit monte carlo case, both ordinary importance sampling and weighted importance sampling are biased but eventually converge to zero.

We will look at the implementation of off-policy every visit monte carlo prediction algorithm.

we are estimating

$$Q_\pi(s,a) = \frac{\sum_{t\in T(s,a)} \rho_{t:T(t)-1} G_t}{\sum_{t\in T(s,a)} \rho_{t:T(t)-1}}$$

we approximate it with the weight parameter W_k.

$$Q_\pi(s,a) = \frac{\sum_{t\in T(s,a)} W_k G_t}{\sum_{t\in T(s,a)} W_k}$$

An incremental update rule for $Q_\pi(s,a)$ is

$$Q_{n+1}(s,a) = Q_n(s,a) + \frac{W_n}{C_n}\left[G_n - V_n\right]$$

where $C_{n+1} = C_n + W_{n+1}$

Off policy Every Visit Monte Carlo Prediction Algorithm

Algorithm 6 Off policy Every Visit Monte Carlo Prediction Algorithm

Input an arbitrary policy π and a behaviour policy b
Initialize $Q(s,a)$ arbitrarily for all $s \in S$ and $a \in A(s)$
Initialize $C(s,a)$ with zeros for all $s \in S$ and $a \in A(s)$
while each episode **do**
    Generate an episode $S_0, A_0, R_1, S_1, A_1, R_2, \cdots, S_{T-1}, A_{T-1}, R_T$ under behaviour policy b
    $G \leftarrow 0$
    $W \leftarrow 1$
    for $t = T-1$ to 0 **do**
        $G \leftarrow \gamma G + R_{t+1}$
        $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
        $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$
        $W \leftarrow W \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$
    end for
end while

we now the algorithm to evaluate a state value function under a target policy π and any other behaviour policy b. To improve the policy π, we follow the general policy improvement approach by acting greedy.

$$\pi'(s) = \arg \max_{a \in A(s)} Q_\pi(s,a)$$

such that $\pi'(s) \geq \pi(s)$ for all $s \in S$.
(Off policy Every Visit Monte Carlo Control Algorithm)

Algorithm 7 Off policy Every Visit Monte Carlo Control Algorithm

Initialize $Q(s,a)$ arbitrarily for all $s \in S$ and $a \in A(s)$
Initialize $C(s,a)$ with zeros for all $s \in S$ and $a \in A(s)$
Input an arbitrary soft behaviour policy b ▷ e.g. ϵ-soft
$\pi \leftarrow argmax_a Q(s,a)$
while each episode **do**
    Generate an episode $S_0, A_0, R_1, S_1, A_1, R_2, \cdots, S_{T-1}, A_{T-1}, R_T$ under behaviour policy b
    $G \leftarrow 0$
    $W \leftarrow 1$
    for $t = T-1$ to 0 **do**
        $G \leftarrow \gamma G + R_{t+1}$
        $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
        $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$
        $\pi'(s) \leftarrow \arg\max_{a \in A(s)} Q_\pi(s,a)$
        $\pi \leftarrow \pi'$
        if $A_t \neq \pi(S_t)$ **then**
            Break
        end if
    end for
    $W \leftarrow W \frac{1}{b(A_t|S_t)}$
end while