Dhruv Awasthi: MS2022010

Technical Report for Race Track Problem in Reinforcement Learning

On-Policy Monte Carlo Control:

On-policy methods attempt to evaluate or improve the policy that is used to make decisions. In on-policy control methods the policy is generally soft, meaning that $\Pi(a|s) > 0$ for all $s \in S$ and all $a \in A(s)$, but gradually shifted closer and closer to a deterministic optimal policy.

The algorithm used for On Policy Monte Carlo Method Control is:

On-policy first-visit MC control (for ε-soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

$\pi \leftarrow$ an arbitrary ε-soft policy

$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

$A^* \leftarrow \arg\max_a Q(S_t, a)$ (with ties broken arbitrarily)

For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$

- In on policy method there is only a single policy that is used for sampling the episodes as well as to select the best action, called the target policy.
- This policy is continuously evaluated and improved to generate the optimal policy.
- Here also the target policy is a soft policy that means most of the times it selects the greedy action given a state, but with some probability, say epsilon, it selects a non-greedy action. This helps balance exploration vs exploitation.
- There are two possible variants of this - the first visit method MC control and the every visit.
- To understand the difference between the two variants, let's go through an example.
- In particular, suppose we wish to estimate $v_\Pi(s)$, the value of a state s under policy Π, given a set of episodes obtained by following Π and passing through s. Each

occurrence of state s in an episode is called a visit to s. Of course, s may be visited multiple times in the same episode; let us call the first time it is visited in an episode the first visit to s. The first-visit MC method estimates $v_{\Pi}(s)$ as the average of the returns following first visits to s, whereas the every-visit MC method averages the returns following all visits to s.

- Both first-visit MC and every-visit MC converge to $v_{\Pi}(s)$ as the number of visits (or first visits) to s goes to infinity.

Off-Policy Monte Carlo Control:

Off-policy methods evaluate or improve a policy different from that used to generate the data. All learning control methods face a dilemma: They seek to learn action values conditional on subsequent optimal behavior, but they need to behave non-optimally in order to explore all actions (to find the optimal actions). How can they learn about the optimal policy while behaving according to an exploratory policy? The on-policy approach in the preceding section is actually a compromise—it learns action values not for the optimal policy, but for a near-optimal policy that still explores. A more straightforward approach is to use two policies, one that is learned about and that becomes the optimal policy, and one that is more exploratory and is used to generate behavior. The policy being learned about is called the target policy, and the policy used to generate behavior is called the behavior policy. In this case we say that learning is from data "off" the target policy, and the overall process is termed off-policy learning.

Off-policy methods are often of greater variance and are slower to converge, but are more powerful and general.

The algorithm used for Off Policy Monte Carlo Method Control is:

Off-policy MC control, for estimating $\pi \approx \pi_*$

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
$Q(s,a) \in \mathbb{R}$ (arbitrarily)
$C(s,a) \leftarrow 0$
$\pi(s) \leftarrow \operatorname{argmax}_a Q(s,a)$ (with ties broken consistently)

Loop forever (for each episode):
$b \leftarrow$ any soft policy
Generate an episode using b: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
$G \leftarrow 0$
$W \leftarrow 1$
Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
$G \leftarrow \gamma G + R_{t+1}$
$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$
$\pi(S_t) \leftarrow \operatorname{argmax}_a Q(S_t, a)$ (with ties broken consistently)
If $A_t \neq \pi(S_t)$ then exit inner Loop (proceed to next episode)
$W \leftarrow W \frac{1}{b(A_t|S_t)}$

- The off policy method makes use of two policies -
  - The policy used to generate behavior i.e. episodes, is called the behavior policy, and
  - The policy used to select the best action and one which is continuously evaluated and improved is called the target policy.
- The policy is also a soft policy that means that most of the times it selects the greedy action given a state, but with some probability, say epsilon, it selects a non-greedy action. This helps balance exploration vs exploitation.
- Suppose we wish to estimate v_Π or q_Π , but all we have are episodes following another policy b, where b ≠ Π. In this case, Π is the target policy, b is the behavior policy, and both policies are considered fixed and given.
- In order to use episodes from b to estimate values for Π, we require that every action taken under Π is also taken, at least occasionally, under b. That is, we require that Π(a|s) > 0 implies b(a|s) > 0. This is called the *assumption of coverage*. It follows from coverage that b must be stochastic in states where it is not identical to Π. The target policy Π, on the other hand, may be deterministic.
- Almost all off-policy methods utilize importance sampling, a general technique for estimating expected values under one distribution given samples from another. We apply importance sampling to off-policy learning by weighting returns according to the relative probability of their trajectories occurring under the target and behavior policies, called the importance-sampling ratio.

$$\rho_{t:T-1} \doteq \frac{\prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k,A_k)}{\prod_{k=t}^{T-1} b(A_k|S_k)p(S_{k+1}|S_k,A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}. \tag{5.3}$$

- Recall that we wish to estimate the expected returns (values) under the target policy, but all we have are returns G_t due to the behavior policy. These returns have the wrong expectation $E[G_t|S_t=s] = v_b(s)$ and so cannot be averaged to obtain v_Π . This is where importance sampling comes in. The ratio $p_{t:T-1}$ transforms the returns to have the right expected value:

> Now we are ready to give a Monte Carlo algorithm that averages returns from a batch of observed episodes following policy b to estimate $v_\pi(s)$. It is convenient here to number time steps in a way that increases across episode boundaries. That is, if the first episode of the batch ends in a terminal state at time 100, then the next episode begins at time $t = 101$. This enables us to use time-step numbers to refer to particular steps in particular episodes. In particular, we can define the set of all time steps in which state s is visited, denoted $\mathcal{T}(s)$. This is for an every-visit method; for a first-visit method, $\mathcal{T}(s)$ would only include time steps that were first visits to s within their episodes. Also, let $T(t)$ denote the first time of termination following time t, and G_t denote the return after t up through $T(t)$. Then $\{G_t\}_{t\in\mathcal{T}(s)}$ are the returns that pertain to state s, and $\{\rho_{t:T(t)-1}\}_{t\in\mathcal{T}(s)}$ are the corresponding importance-sampling ratios. To estimate $v_\pi(s)$, we simply scale the returns by the ratios and average the results:
>
> $$V(s) \doteq \frac{\sum_{t\in\mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{|\mathcal{T}(s)|}. \tag{5.5}$$

- When importance sampling is done as a simple average in above way, it is called *ordinary importance sampling.*
- An important alternative is *weighted importance sampling*, which uses a weighted average, defined as:

$$V(s) \doteq \frac{\sum_{t\in\mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{\sum_{t\in\mathcal{T}(s)} \rho_{t:T(t)-1}}, \tag{5.6}$$

or zero is the denominator is zero.

- Ordinary importance sampling is unbiased whereas weighted importance sampling is biased (though the bias converges asymptotically to zero).
- The variance of ordinary importance sampling is in general unbounded because the variance of the ratios can be unbounded, whereas in the weighted estimator the largest weight on any single return is one.
- In practice, the weighted estimator usually has dramatically lower variance and is strongly preferred.

- In practice, every-visit methods are often preferred because they remove the need to keep track of which states have been visited and because they are much easier to extend to approximations.

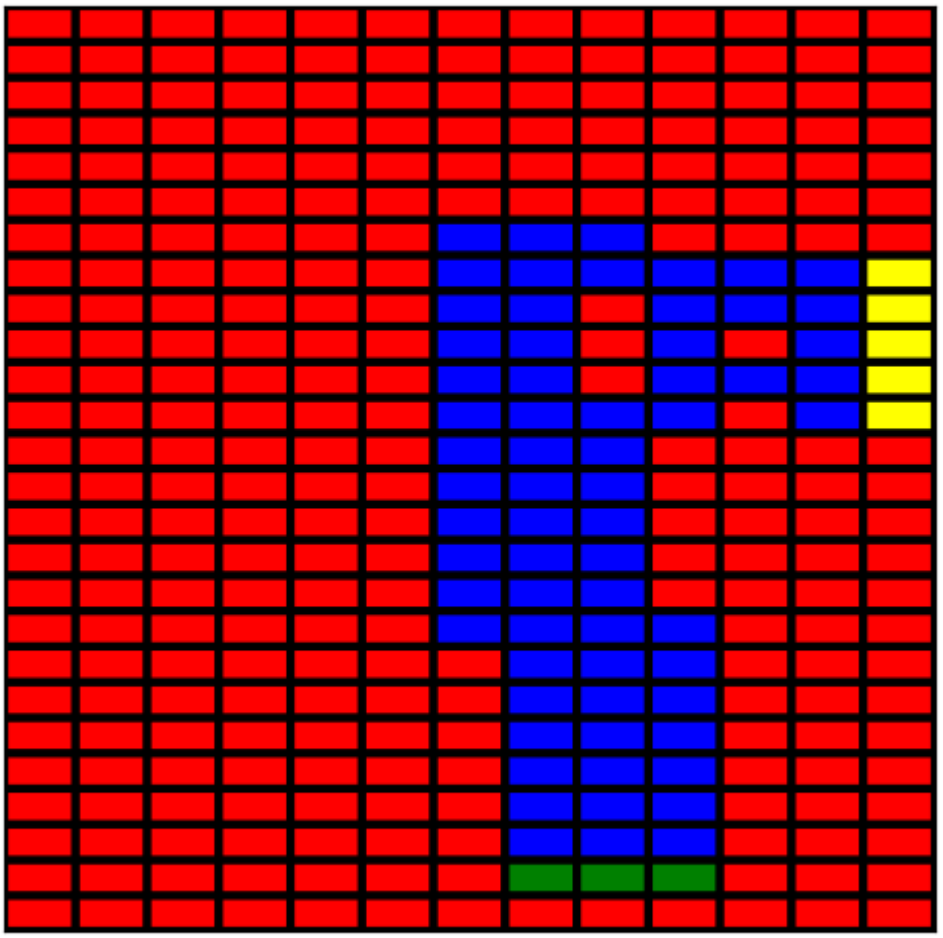
Convergence:

- Off-policy methods are often of greater variance and are comparatively slower to converge than the on-policy methods.
- But off-policy methods are more powerful and general than on-policy methods.

Race Track:

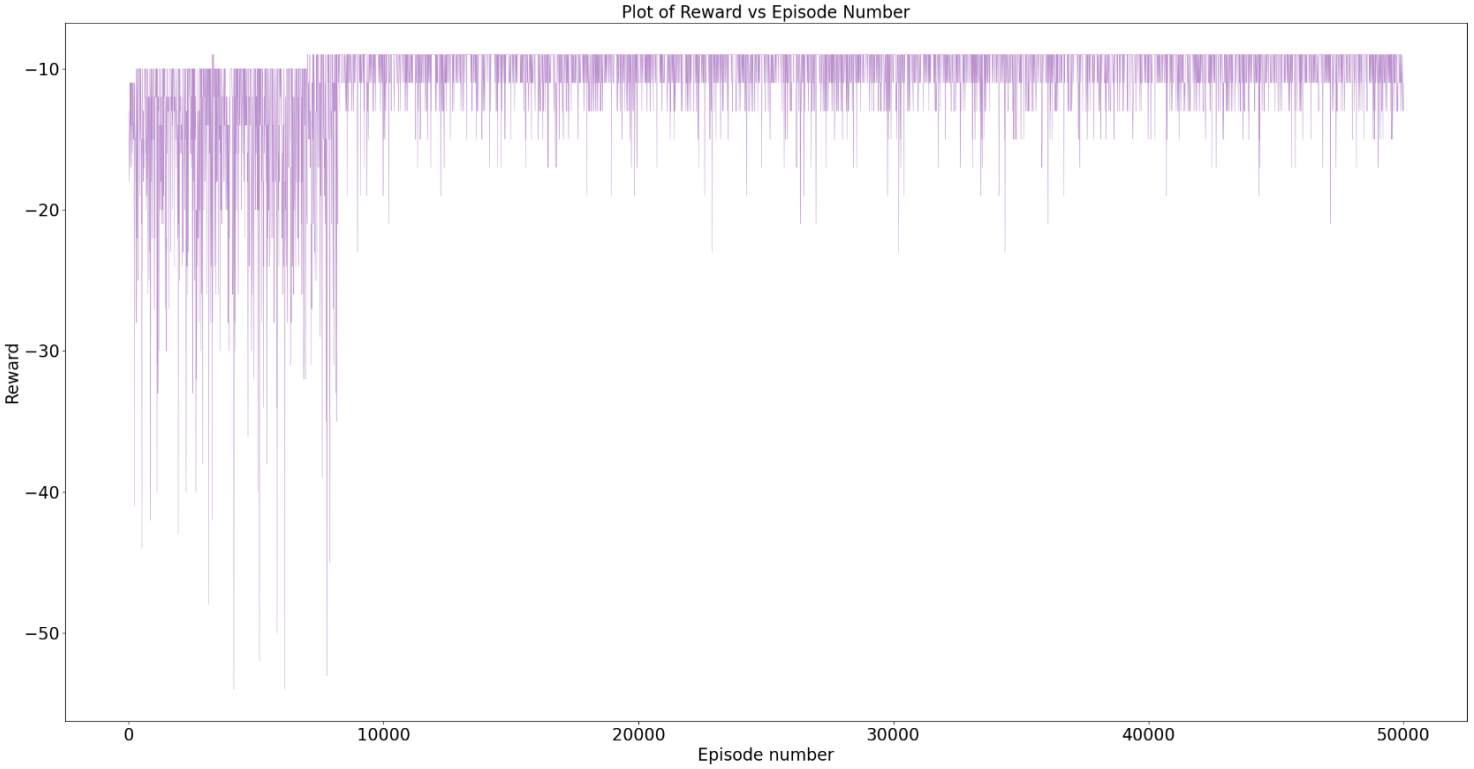
Here is a sample visualization of the race track provided in the assignment.

- All the red blocks mark the invalid positions.
- All the blue blocks mark the valid positions.
- All the green blocks mark the starting positions, and
- All the yellow blocks mark the finishing positions.



Result:

- Here is a sample plot generated while training the agent.
- We are giving a reward of -1 at each time step, and the objective is to maximize those accumulated rewards over total time.
- The plot is a reward vs number of episodes.
- As we can see in the plot, initially the reward was highly negative. This is when we have no which is the best action given a state under a policy.
- Over time, we accumulate that information and it starts increasing.
- And after a point it kind of staturates when we have learned the optimal policy.
- The small decrements we see in the reward even after learning the optimal policy is because of the exploration that we are trying to do. But overall we have learnt an optimal policy that works really well.


Plot of Reward vs Episode Number
−10
−20
−30
−40
−50
Reward
0
10000
20000
30000
40000
50000
Episode number