<u>Chapter 5 - Monte Carlo Methods</u> (Aravind S - EE14B013 - RL project RA 2)

Monte Carlo prediction

- In most cases we won't have complete knowledge of the system dynamics (MDP params are not known) but we will be able to interact with the environment. In that case, Monte Carlo methods seem suitable as we obtain an estimate of the value/action-value functions and hence policy from the different episodes (sampled from the environment).
- Following a policy π, episodes are generated. Say, we want to estimate value functions. V(s) is the average of returns observed from the state s. This is true as the expected return is the average of a large number of samples drawn according to the distribution (remember: policy π is followed).
- **Advantages over DP**
  - Optimal behaviour can be learnt just from interaction with the environment.
  - Focussed learning - value functions of a particular state can be estimated separately. This is because to estimate V(s) we need a bunch of episodes starting from state 's' and we need not compute values for other states.
  - No bootstrapping - less Markovian estimate. Practically, systems may be non-markovian. In such cases, MC gives a more reliable estimate whereas DP/TD gives a biased (Markov nature is assumed) estimate.
- **Variants**
  - First visit - Returns from first visit to a state is considered. True estimate of V(s), but may not be reliable is 's' is a rare state for the policy π.
  - Every visit - Returns from multiple visit to a state is considered. A biased estimate, but we obtain a relatively large number of samples for a rare state 's'.

Monte Carlo estimation of action-values

- The action-value function is estimated using returns obtained when a particular (state, action) pair is visited.
- Both first-visit and every-visit variants can be used depending on the type of episodes available.

Monte Carlo control (on-policy)

- To approximate optimal policies, we use GPI. We evaluate action-values rather than values because we don't know the dynamics of the system and hence estimation of policies from values isn't possible (Bellman equation).
- Action-values are estimated for an estimated π. π is computed greedily w.r.t current estimate of action-values. This is repeated until convergence (Each (s, a) is visited infinitely often).
- The problem is when a fixed policy π is followed, some (state, action) pairs will never occur. This problem is tackled using:
  - Exploring start s (ES) - Episodes are generated s.t the initial (s, a) covers all possible combinations and hence when infinite episodes are obtained, the estimates converge to true action-value functions.
  - Without ES - ES is unlikely as it can't be simulated practically. The alternate approach is to use ε-soft policies (page 109 - para 1). The disadvantage with this method is the optimal policy obtained is the best among all the possible ε-soft policies. This may not be the overall best optimal policy.
- Policy evaluation can be implemented in an incremental fashion using cumulative sum of weights for the states encountered (Section 5.6).

Off-policy prediction via Importance sampling (page 112 - detailed explanation of importance sampling)

- The problem with MC methods is - you need to obtain samples following the policy π. But for your estimate to be reliable you need to explore.
- This is solved using off-policy methods where you deal with 2 policies - behaviour policy (used to generate episodes) & target policy (the policy which is learnt). We generate the episodes following a policy μ and estimate values for policy π.
- 2 things to note:
  - Coverage - μ should be stochastic and hence have a non-zero probability of selecting actions which might be selected when we follow π.
  - Ordinary (unbiased, higher variance [sometimes infinite]) Vs Weighted (biased, lower variance) - Estimates from weighted importance sampling are more reliable even though the estimates don't correspond to an expectation over policy π. But given a large number of episodes both these estimates are close enough and hence converge to the same true optimal value.

Off-policy Monte Carlo control

- As expected, Monte Carlo control is performed using GPI. Prediction step is done using importance sampling (policy evaluation following some behaviour policy, μ with an estimate of target policy, π).
- The policy improvement step is a greedification step. π is estimated greedily with the current estimate of action-values.
- μ is preferably an ε-soft policy to make sure sufficient exploration is done. The problem with this is learning occurs when part of the trajectory is obtained from a greedy policy. If there are lot of exploratory actions, learning is slow and hence it takes a long time converge. This situation can be avoided by using methods like temporal difference learning.

Return-specific importance sampling

- Discount γ - measure of probability of termination. Now a flat partial return up to a horizon h (Page 121) is defined. Following the derivation in page 121, we obtain a different importance sampling based estimator.
- The advantage of this is, for a discounted return setup, later rewards contribute less to the return and hence the weights in the ordinary importance sampling estimator result in higher variance in the estimate.
- Per-reward importance sampling - importance-sampling weights are simplified resulting in a low-variance estimate (derivation in page 122).