Monte Carlo Methods
Week #5

Introduction

- *Monte Carlo (MC) Methods*
  - do not assume complete knowledge of environment (unlike DP methods which assume a perfect environment model)
  - require experience
    - *sample sequences of states, actions, and rewards from interaction (on-line or simulated) with an environment*
- *on-line experience* exposes agent to *learning in an unknown environment* whereas *simulated experience* does require a model, but the *model needs only generate sample transitions and not the complete probability distribution*.
- In *MC* methods, the *unknown value functions* are computed from *averaging sample returns*.

Set up

- We assume the environment is a finite MDP.
- Finite set of states: *S*
- Finite set of actions: *A(s)*
- Dynamics of environment given by:
  - a set of transition probabilities, and
  - $P^a_{s_i s_j} = P(s_{t+1} = s_j \mid s_t = s_i, a_t = a)$ ***unknown***
  - the expected immediate reward
  - $R^a_{s_i s_j} = E(r_{t+1} \mid s_t = s_i, a_t = a, s_{t+1} = s_j)$ ***unknown***
- for all $s_i \in S$, $s_j \in S^+$ and a $\in A(s)$.

MC Policy Evaluation

- *Value of a state:* *expected cumulative future discounted reward*
- To estimate *state value* from *experience*, *returns observed after visits to that state should be averaged*.
- *Every-visit MC method*: method of estimating $V^{\pi}(s)$ as the average of the returns following all visits to the state *s* in a set of episodes.
- *First-visit MC method*: method of estimating $V^{\pi}(s)$ as the average of the returns following first visits to the state *s* in a set of episodes.

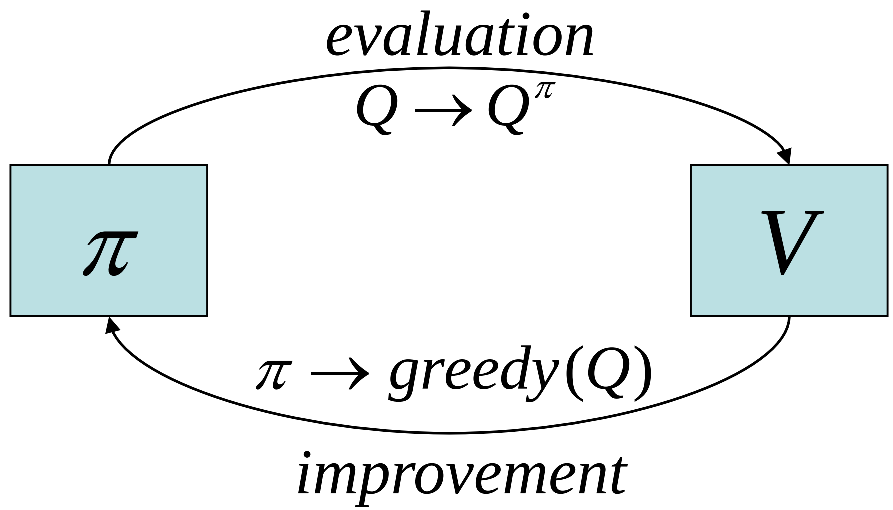
First-visit MC Method

- *Initialize:*
- π*: the policy to be evaluated*
- *V: an arbitrary state-value function*
- *Returns(s): an empty list, for all s∈S*
- *Repeat forever:*
  - *Generate an episode using* π
  - *For each state s appearing in the episode:*
    - *R* ← *return following the first occurrence of s*
    - *Append R to Returns(s)*
    - *V(s)* ← *average(Returns(s))*

MC Estimation of Action Values

- For environments without a model, state values alone are not sufficient to determine a policy. A *one-step look-ahead* operation is enough to find the action that leads to the best combination of reward and next state.
- We need to estimate the value of the state-action pair, $Q^{\pi}(s,a)$, the expected return when starting in state *s*, taking action *a*.
- *Problem*: many state-action pairs may never be visited.
  - We need to *maintain exploration.*
  - *Two assumptions* to solve the problem*:*
    - *Exploring starts: each state-action pair has a non-zero probability.*
    - *Infinite number of episodes: all state-action pairs are visited infinitely many times*

Generalized Policy Iteration in MC Methods



$$\vdots$$

$$\pi^{*} \leftrightarrow Q^{*}$$

where π and Q* denote optimal policy and action values, respectively.*

Policy Iteration: MC Version

$$\pi_0 \xrightarrow{E} Q^{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} Q^{\pi_1} \xrightarrow{I} \cdots \pi_{opt} \xrightarrow{E} Q^{\pi_{opt}}$$

In the above diagram denoting the policy iteration the transition *E* and *I* stand for the policy evaluation and policy improvement phases of the policy iteration steps in respective order [1]. For any *Q*, the corresponding *greedy policy* selects an *action with* the *maximal Q-value* as in the following:

$$\pi(s) = \arg\max_a Q(s,a)$$

The figure is an exact replication of the figure on page 97 of [1].

Removing the Second Assumption...

- To remove the assumption of *infinite number of episodes* we again use the same idea of *value iteration* (i.e., instead of many just a single iteration of policy evaluation between each policy improvement step) as we did during the discussion of dynamic programming.
- An extreme version is the *in-place value iteration* where at each iteration only one state is updated.

Monte Carlo ES Method

- *Initialize π(s) and Q(s,a) arbitrarily for all $s \in S$, $a \in A(s)$:*
- *Returns(s,a)* $\leftarrow$ *empty list*
- *Repeat forever:*
  - *Generate an episode using exploring starts and* π
  - *For each pair (s,a) appearing in the episode:*
    - *R* $\leftarrow$ *return following the first occurrence of s,a*
    - *Append R to Returns(s,a)*
    - *Q(s,a)* $\leftarrow$ *average(Returns(s,a))*
  - *For each s in the episode:*

$$\pi(s) \leftarrow \arg\max_a Q(s,a)$$

Removing the first Assumption...

- To remove the assumption of *exploring starts* the agent needs to continually select actions.

- Two types of MC control to select actions are:
  - *on-policy* MC control, and
  - *off-policy* MC control.

On-Policy MC Control

- In *on-policy* control, the policy we use to select actions (behavioral policy) is the same as the policy we use to estimate action values (estimation policy);
  - i.e., ***behavioral policy ≡ estimation policy***
- Idea is that of GPI.
- Any policy we discussed in the second week may be used in on-policy MC control.
- Without ES, the policy moves to an ε-greedy one.

On-Policy MC Control Algorithm

- *Initialize for all $s \in S$, $a \in A(s)$:*
- *$Q(s,a) \leftarrow$ arbitrary*
- *$Returns(s,a) \leftarrow$ empty list*
- *Repeat forever:*
  - *Generate an episode using π*
  - *For each pair (s,a) appearing in the episode:*
    - *$R \leftarrow$ return following the first occurrence of s,a*
    - *Append R to $Returns(s,a)$*
    - *$Q(s,a) \leftarrow average(Returns(s,a))$*
  - *For each s in the episode:*

$$a^* \leftarrow \arg\max_a Q(s,a)$$

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

Off-Policy MC Control

- In *off-policy* control, the policy we use to select actions (*behavior policy*), π_b, and the policy we use to estimate action values (*estimation policy*), π_e, are separated.
- The question here is, how the state value estimates will be correctly updated using the estimation policy to reflect the selections of the behavior policy.

Off-Policy MC Control… (2)

- First requirement to work this out is that actions taken in π_e are also taken in π_b.
- Second, the rates of possibility of occurrence of any sequence between π_e and π_b starting from a specific visit to state s should be balanced.
- That is, consider i^{th} first visit to state s and the complete sequence of states and actions following that visit. We define $p_{i,e}(s)$ and $p_{i,b}(s)$ as the probability of occurrence of the sequence mentioned above given policies π_e and π_b, respectively. Further, let $R_i(s)$ denote the corresponding observed return from state s, and $T_i(s)$ be the time of termination of the i^{th} episode involving state s. Then,

$$p_i(s_t) = \prod_{k=t}^{T_i(s)-1} \pi(s_k, a_k) P_{s_k s_{k+1}}^{a_k}$$

Off-Policy MC Control... (3)

- Assigning weights to each return by its relative probability of occurrence under π_e and π_b, $p_{i,e}(s)/p_{i,b}(s)$, the value estimate can be obtained by

$$V(s) = \frac{\sum_{i=1}^{n_s} \frac{p_{i,e}(s)}{p_{i,b}(s)} R_i(s)}{\sum_{i=1}^{n_s} \frac{p_{i,e}(s)}{p_{i,b}(s)}}$$

- where n_s is the number of returns observed from state s.

Off-Policy MC Control... (4)

- Looking below at the ratio of the probability we see that it depends upon the policies and not at all on the environment's dynamics.

$$\frac{p_{i,e}(s_t)}{p_{i,b}(s_t)} = \frac{\prod_{k=t}^{T_i(s)-1} \pi_e(s_k, a_k) P_{s_k s_{k+1}}^{a_k}}{\prod_{k=t}^{T_i(s)-1} \pi_b(s_k, a_k) P_{s_k s_{k+1}}^{a_k}} = \frac{\prod_{k=t}^{T_i(s)-1} \pi_e(s_k, a_k)}{\prod_{k=t}^{T_i(s)-1} \pi_b(s_k, a_k)}$$

Off-Policy MC Control Algorithm

- *Initialize for all $s \in S$, $a \in A(s)$:*
- *$Q(s,a) \leftarrow$ arbitrary*
- *$N(s,a) \leftarrow 0$;* //Numerator
- *$D(s,a) \leftarrow 0$;* //Denominator of $Q(s,a)$
- *Repeat forever:*
  - *Select a policy π_e and use it to generate an episode*
    - $s_0, a_0, r_1, s_1, a_1, r_2, s_2, a_2, r_3, \ldots, s_{T-1}, a_{T-1}, r_T, s_T.$
  - $\tau \leftarrow$ *latest time at which* $a_\tau \neq \pi_b(s_\tau)$

//...continues on the next page

Off-Policy MC Control Algorithm

- *For each pair (s,a) appearing in the episode at time τ or later:*
  - *t ← the time of first occurrence of s,a such that t≥ τ*

$$w \leftarrow \prod_{k=t+1}^{T-1} \frac{1}{\pi_b(s_k, a_k)}$$

$$N(s,a) \leftarrow N(s,a) + wR_t$$

$$D(s,a) \leftarrow D(s,a) + w$$

$$Q(s,a) \leftarrow \frac{N(s,a)}{D(s,a)}$$

- *For each s ∈ S:*

$$\pi(s) \leftarrow \arg\max_a Q(s,a)$$

References

- [1] Sutton, R. S. and Barto A. G., "*Reinforcement Learning: An introduction*," MIT Press, 1998