# Monte Carlo Prediction

Thursday, March 26, 2020 10:15 PM

Reinforcement Learning

Monte Carlo Prediction

Sampling.

Expirence Samples.

**Monte Carlo Prediction**

$S_0, A_0, R_1, S_1, A_1, R_2, \ldots$

Averaging. sample returns.

$\pi \longrightarrow v_\pi$

First visit

Every visit

→ Ep1 → $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, S_3, \ldots\ldots S_{t-1}, A_{t-1}, R_t, S_t$

$S_x$ $S_x$

→ Ep2 →

→ Ep3 →

$v_\pi(s) =$

→ Ep 3 →

$v_\pi(s) =$

$S_x$

## Blackjack example:

D

P

- Objective of the game is to obtain cards, sum of whose numerical values is as great as possible without exceeding 21.
- All face cards (King, Queen, Jack), have a numerical value of 10.
- Ace can either be 1 or 11.
- Number cards have value equal to its number
- There is dealer the player is playing with.
- The game begins with two cards dealt to both dealer and player. One of the dealers card is face up and another is face down. If player has 21 immediately (a 10 and an Ace), its caller *natural.* He then wins, unless dealer also has a natural, in which case, it is draw. If the player does not have a natural, then he can request additional cards one by one *(hits)*, until he either stops *(sticks)* or exceeds *21 (goes bust)*. If player goes bust, he loses; if he sticks then it becomes the dealer's turn. The dealer hits or sticks according to some strategy. If the dealer goes bust, then the player wins; otherwise the outcome - win, lose or draw is determined by whose final sum is closer to 21.

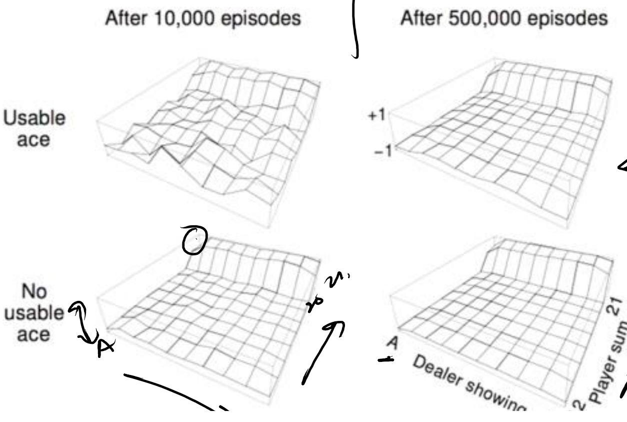
- Modeling it as MDP:
  - Each game can be seen as an episode.
  - Our player is the agent and the dealer can be seen as the component of environment. For ease of explanation, we'll fix the strategy for the dealer. The dealer sticks on any sum of 17 or greater, and hits otherwise. Since the dealer is part of the environment, the agent doesn't have access to this strategy directly.
  - Rewards of +1, -1 or 0 given at the end of the episode, for a win, lose or draw respectively.
  - We assume that the cards are drawn from an infinite deck, so there's no advantage to keeping track of cards already dealt.
  - The episodes are undiscounted.
  - The state is made of three variables: Y 21
    - The current sum of agent (12-21) → 10
    - Dealer's one showing card (Ace-10) → 10
    - Usable Ace (Yes or No) [If player holds an ace which can be used as 11 without going bust then it is called an usable ace]. → 2
  - Actions can be:
    - Hit (Request a card)
    - Stick (Stop requesting card. Give turn to the dealer)

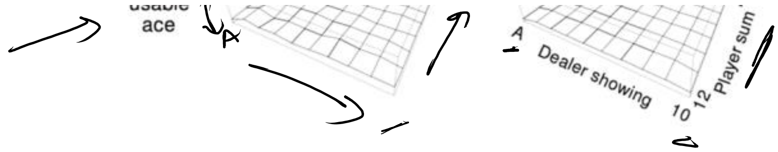
$|S| = 10 \times 10 \times 2 = 200$

- Policy to be evaluated:
  - **The player sticks if sum is 20 or 21, otherwise hits.**

6 A → 11 } 17

→ 6 9 A → 1 } 26 > 21






**Monte Carlo prediction (First visit):**

$S_0 \to 1$

$S_1 \to 1$

$S_2 \to \boxed{R_4}$

$S_3 \to \ldots$

**First-visit MC prediction, for estimating $V \approx v_\pi$**

Input: a policy $\pi$ to be evaluated
Initialize:
  $V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$
  $Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Loop forever (for each episode):
  Generate an episode following $\pi$: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$
  $G \leftarrow 0$
  Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
    $G \leftarrow G + R_{t+1}$
    Unless $S_t$ appears in $S_0, S_1, \ldots, S_{t-1}$:
      Append $G$ to $Returns(S_t)$
      $V(S_t) \leftarrow \text{average}(Returns(S_t))$


$$\boxed{G_t = R_{t+1} + \gamma G_{t+1}}$$

$$G_3 = R_4 + \gamma G_4$$

$$G_4 = R_5 + \gamma G_5$$

$$G_5 = 0$$

MC control - Exploring starts

Sunday, March 29, 2020 3:02 PM

## Reinforcement Learning

### Monte Carlo Control - Exploring Starts

$q_\pi(s,a)$ $q_*$

**Estimating action value functions:**

$q_\pi(s,a)$

$s,a \to \pi$

$\to S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, S_3, \ldots$

$S_x$

$S_x, A_x,$

Exploration

$\pi(s) \to a_1$

$(s, a_1)$ ✓

$s \to a_1, a_2, a_3$

$(s, a_1)$ ✓ $\quad s \rightarrow a_2$, $s \rightarrow a_3$

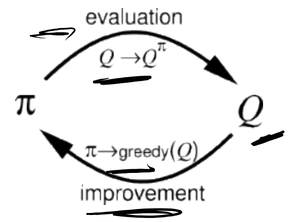
$(s, a_2)$ ×

$(s, a_3)$ ×

**Monte Carlo Control:**

$$\pi_0 \xrightarrow{E} q_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} q_{\pi_1} \xrightarrow{I} \pi_2 \rightarrow \cdots \xrightarrow{E} q_* \xrightarrow{I} \pi_*$$

$$\pi_k \rightarrow q_{\pi_k}$$




**Monte Carlo Control (using Exploring Starts) algorithm:**

**Monte Carlo ES (Exploring Starts), for estimating $\pi \approx \pi_*$**

Initialize:
- $\pi(s) \in \mathcal{A}(s)$ (arbitrarily), for all $s \in \mathcal{S}$
- $Q(s, a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$
- $Returns(s, a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Loop forever (for each episode):
- Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}(S_0)$ randomly such that all pairs have probability $> 0$
- Generate an episode from $S_0, A_0$, following $\pi$: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
- $G \leftarrow 0$
- 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 $Returns(S_t, A_t)$
    - $Q(S_t, A_t) \leftarrow$ average$(Returns(S_t, A_t))$
    - $\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

$S_0, a_0 \rightarrow \square$

$S_0, a_1 \rightarrow \square$

$S_0, a_3 \rightarrow \square$

$\vdots$

$S_1, a_0 \rightarrow \square$

$S_2, a_1 \rightarrow \square$

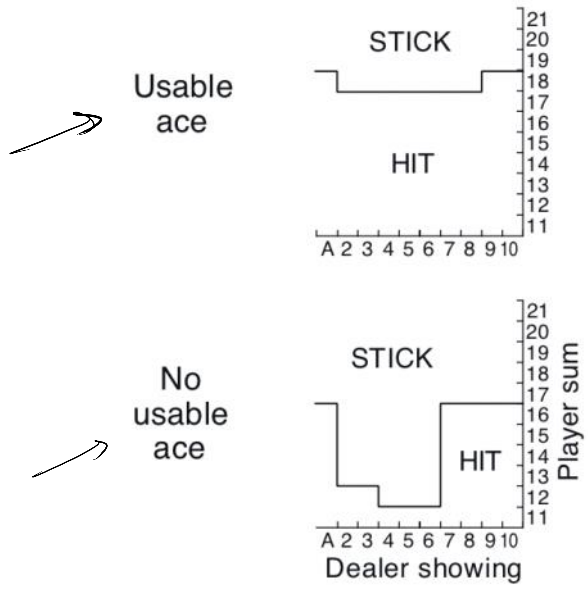
$\vdots \quad \rightarrow \square$

## **Blackjack example:**

- Objective of the game is to obtain cards, sum of whose numerical values is as great as possible without exceeding 21.
- All face cards (King, Queen, Jack), have a numerical value of 10.
- Ace can either be 1 or 11.
- Number cards have value equal to its number
- There is dealer the player is playing with.
- The game begins with two cards dealt to both dealer and player. One of the dealers card is face up and another is face down. If player has 21 immediately (a 10 and an Ace), its caller *natural.* He then wins, unless dealer also has a natural, in which case, it is draw. If the player does not have a natural, then he can request additional cards one by one *(hits)*, until he either stops *(sticks)* or exceeds *21 (goes bust)*. If player goes bust, he loses; if he sticks then it becomes the dealer's turn. The dealer hits or sticks according to some strategy. If the dealer goes bust, then the player wins; otherwise the outcome - win, lose or draw is determined by whose final sum is closer to 21.

- Modeling it as MDP:
  - Each game can be seen as an episode.
  - Our player is the agent and the dealer can be seen as the component of environment. For ease of explanation, we'll fix the strategy for the dealer. The dealer sticks on any sum of 17 or greater, and hits otherwise. Since the dealer is part of the environment, the agent doesn't have access to this strategy directly.
  - Rewards of +1, -1 or 0 given at the end of the episode, for a win, lose or draw respectively.
  - We assume that the cards are drawn from an infinite deck, so there's no advantage to keeping track of cards already dealt.
  - The episodes are undiscounted.
  - The state is made of three variables:
    - The current sum of agent (12-21)
    - Dealer's one showing card (Ace-10)
    - Usable Ace (Yes or No) [If player holds an ace which can be used as 11 without going bust then it is called an usable ace].
  - Actions can be:
    - Hit (Request a card)
    - Stick (Stop requesting card. Give turn to the dealer)

- Initial policy:
  - The player sticks if sum is 20 or 21, otherwise hits.



## Reinforcement Learning
### Monte Carlo Control - Using $\epsilon$-soft policies

**Monte Carlo control using $\epsilon$-soft policies:**

Greedy policy $\xrightarrow{95\%}$ optimal Action

Greedy policy $\xrightarrow[5\%]{}$ Non optimal action

$\frac{\epsilon}{|A|}$

$5 \to 1$ G

$5 \to 4$ NG. $\left(\frac{\epsilon}{5} + 4\frac{\epsilon}{5}\right)$

$1 - \epsilon + \left(\frac{\epsilon}{5}\right)$

$\approx 1$

Non greedy actions $\to \frac{\epsilon}{|A|}$

greedy action $\to \underbrace{1 - \epsilon} + \frac{\epsilon}{|A|}$

**Monte Carlo Control (using $\epsilon$-soft policies) algorithm:**

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

Algorithm parameter: small $\varepsilon > 0$

Initialize:

  $\pi \leftarrow$ an arbitrary $\varepsilon$-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 $\pi$: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$

  $G \leftarrow 0$

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

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

$S_0, a_0 \rightarrow$

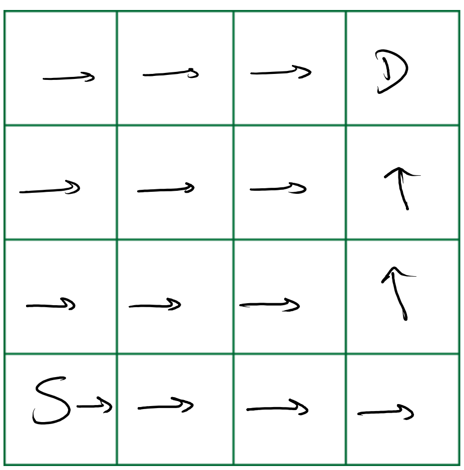
$S_0, a_1 \rightarrow$

$S_0, a_2 \rightarrow$

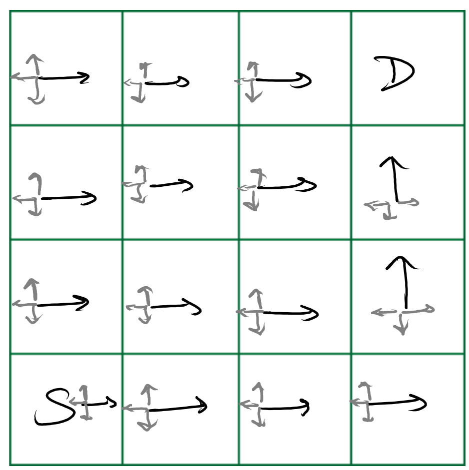
$S_1, a_0 \rightarrow$

$S_1, a_1 \rightarrow$

Deterministic




→ Soft policy.



Off-policy learning

Sunday, March 29, 2020 3:53 PM

## Reinforcement Learning
### Off-Policy learning

**On-Policy Learning:**

**Off-Policy Learning:**

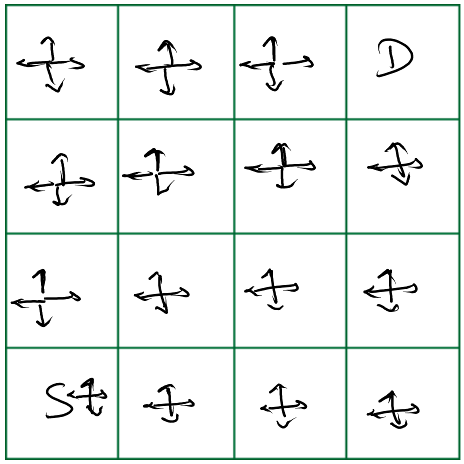
→ Value functions

target policy → $\pi(s|a)$

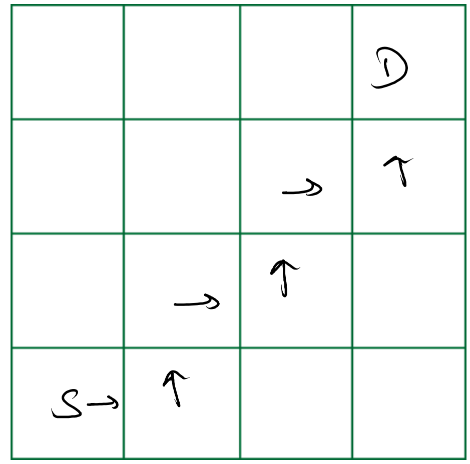
behaviour policy → $b(s|a)$

→ Action which the agent takes

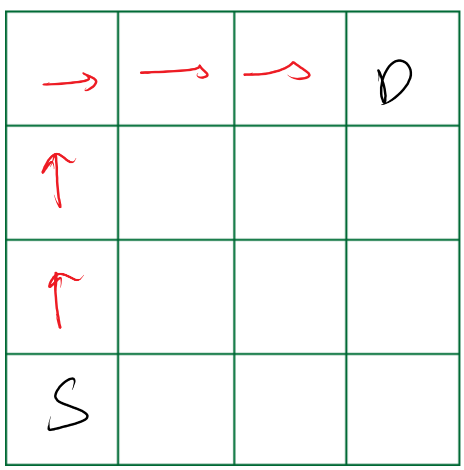
$\Rightarrow b(s|a)$



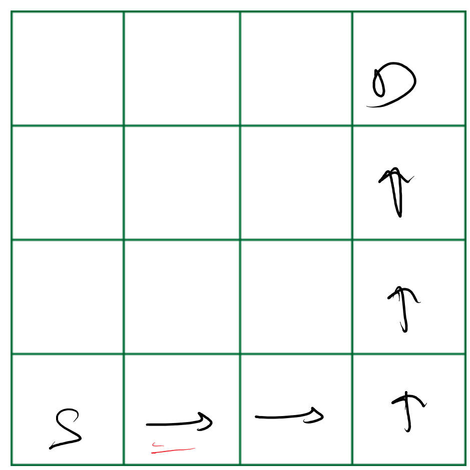

$\rightarrow \underline{\pi(s|a)}$




$b(s|a)$




$\pi(s|a)$




$\pi(s|a) > 0$

$\Rightarrow b(s|a) > 0$

# Importance Sampling

Monday, March 30, 2020 11:43 PM

Reinforcement Learning
Importance Sampling

**Why Importance Sampling?**

$b \sim S_0, A_0, R_1, S_1, A_1, R_2, S_2 \ldots\ldots$

$V_\pi(s) =$

$V_\pi(s) = E_\pi[G_t | S_t = s]$

$f(x)$ $g(x)$

$f(x)$ $\qquad$ $g(x)$

## Derivation of Importance Sampling

Samples : $x \sim b$

$\mathbb{E}_\pi[X]$

$$\mathbb{E}_\pi[X] = \sum_{x \in X} x \cdot \pi(x) = \sum_{x \in X} x \cdot \left(\frac{\pi(x)}{b(x)}\right) \cdot b(x)$$

$\rho(x)$

$$= \sum_{x \in X} x \cdot \rho(x) \cdot b(x)$$

$$= \mathbb{E}_b[X\rho(X)]$$

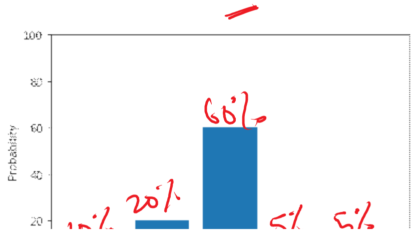
$$= \frac{1}{n}\sum_{i=1}^{n} x_i \rho(x_i)$$

$x \sim b$

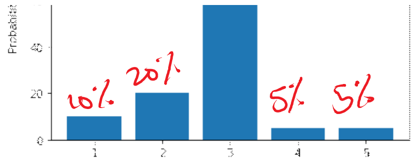
*Sample* : $x \sim b$
*Estimate* : $\mathbb{E}_\pi[X]$

$$\mathbb{E}_\pi[X] = \sum_{x \in X} x\pi(x)$$
$$= \sum_{x \in X} x\pi(x)\frac{b(x)}{b(x)}$$
$$= \sum_{x \in X} x\frac{\pi(x)}{b(x)}b(x)$$
$$= \sum_{x \in X} x\rho(x)b(x) \qquad , \qquad \rho(x) = \frac{\pi(x)}{b(x)} = \textit{Importance Sampling ratio}$$
$$= \mathbb{E}_b[X\rho(X)]$$
$$= \tfrac{1}{n}\textstyle\sum_{i=1}^{n} x_i\rho(x_i) \qquad , \qquad n = \textit{number of samples}$$

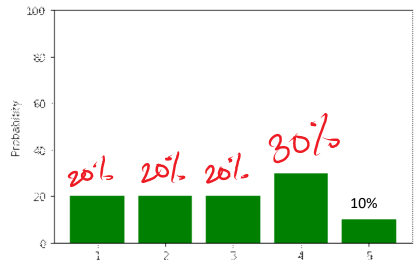
$\pi(x)$




$\mathbb{E}_\pi[X] = 2.75$




$b(x)$




$X \sim b : [4, 4, 1, 3, 1]$

$$\rho(4) = \frac{\pi(4)}{b(4)} = \frac{5}{30} = \frac{1}{6}$$

$$\rho(1) = \frac{\pi(1)}{b(1)} = \frac{10}{20} = \frac{1}{2}$$

$$\rho(3) = \frac{\pi(3)}{b(3)} = \frac{60}{20} = 3$$

$$\frac{1}{5}\left[4 \cdot \frac{1}{6} + 4 \cdot \frac{1}{6} + 1 \times \frac{1}{2} + 3 \times 3 + 1 \times \frac{1}{2}\right]$$

$$= \frac{1}{5}\left[\frac{2}{3} + \frac{2}{3} + \frac{1}{2} + 9 + \frac{1}{2}\right]$$

$$= \frac{1}{5}\left[\frac{4}{3} + 9 + \frac{1}{2}\right]$$

$$\approx 2.26$$

Off-policy control and Prediction

Thursday, April 2, 2020 10:21 PM

Reinforcement Learning

Off-policy prediction and control

**Off-policy learning using importance sampling:**

$x \sim b(x)$

$\pi(x)$

$E_\pi[X] = E_b[X \cdot \rho(X)]$ $\rho(x) = \frac{\pi(x)}{b(x)}$

$= \frac{1}{n}\sum_{i=1}^{n} x_i \cdot \rho(x_i)$ $x_i \sim b$

$v_\pi(s) = E_\pi[G_t \mid S_t = s]$

$S_t$, $A_t, S_{t+1}, A_{t+1}, S_{t+2}, A_{t+2} \cdots S_T$

$Pr\{A_t, S_{t+1}, A_{t+1}, S_{t+2}, A_{t+2} \cdots S_T \mid S_t, A_{t:T-1} \sim \pi\}$

$Probability\ of\ some\ state-action\ trajectory\ A_t, S_{t+1}, A_{t+1}, \ldots, S_T, when\ starting\ from\ some\ state\ S_t, according\ to\ policy\ \pi$:

$$P\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T \mid S_t, A_{t:T-1} \sim \pi\} = \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\,\pi(A_{t+1}|S_{t+1})p(S_{t+2}|S_{t+1}, A_{t+1}) \ldots \ldots \pi(A_{T-1}|S_{T-1})p(S_T|S_{T-1}, A_{T-1})$$

$$= \prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k,A_k)$$

*Similarly,*

$$P\{A_t, S_{t+1}, A_{t+1}, \ldots\ldots, S_T | S_t, A_{t:T-1} \sim b\} = \prod_{k=t}^{T-1} b(A_k|S_k)p(S_{k+1}|S_k,A_k)$$

*So the importance sampling ratio for the trajectory would be:*

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

*Hence,*

$$\mathbb{E}[\rho_{t:T-1}G_t|S_t = s] = v_\pi(s)$$

$G_t \sim b$

$E[G_t | S_t = s] = V_b(s)$

$E[\rho_{t:T-1} G_t | S_t = s] = V_\pi(s)$

50

100

101 152

153 551

$J(s)$

$T(t)$

$S$

$G_1, G_2, G_3 \ldots \quad G_{n-1}$

$W_1, W_2, W_3 \ldots \quad W_{n-1}$

*Estimating $v_\pi(s)$ using ordinary importance sampling :*

$$V(s) = \frac{\sum_{t\in J(s)} \rho_{t:T(t)-1} G_t}{|J(s)|}$$

*Estimating $v_\pi(s)$ using weighted importance sampling :*

$$V(s) = \frac{\sum_{t\in J(s)} \rho_{t:T(t)-1} G_t}{\sum_{t\in J(s)} \rho_{t:T(t)-1}}$$

$$V_n = \frac{\sum_{k=1}^{n-1} W_k \cdot G_k}{\sum_{k=1}^{n-1} W_k}$$

$$C_n = \sum^{n} W_k$$

$$V_{n+1} = \frac{\sum^{n} W_k G_k}{\sum W_k} = \frac{W_n G_n + \sum^{n-1}(W_k \cdot G_k)}{\sum^{n} W_k}$$

$$= \frac{W_n G_n + \sum^{n-1}(W_k) \cdot V_n}{\sum^{n} W_k} = \frac{W_n G_n + C_{n-1} V_n}{C_n}$$

$$= \frac{W_n G_n}{C_n} + \frac{C_{n-1} V_n}{C_n} - V_n + V_n = \frac{W_n G_n}{C_n} + V_n + \frac{C_{n-1}V_n - C_n V_n}{C_n}$$

$$= \frac{W_n G_n}{C_n} + V_n - \frac{V_n [C_n - C_{n-1}]}{C_n}$$

$$= \frac{W_n G_n}{C_n} + V_n - \frac{V_n W_n}{C_n}$$

$$= V_n + \frac{W_n}{C_n}[G_n - V_n]$$

$$V_n = \frac{\sum_{k=1}^{n-1} W_k G_k}{\sum_{k=1}^{n-1} W_k} \quad , n \geq 2$$

$$V_n = \frac{\sum_{k=1}^{n-1} W_k G_k}{\sum_{k=1}^{n-1} W_k} \quad , n \geq 2$$

$C_n$ $\qquad$ $C_n$

$$= V_n + \frac{W_n}{C_n}\left[G_n - V_n\right]$$

*Incrementally,*

$$V_{n+1} = V_n + \frac{W_n}{C_n}\left[G_n - V_n\right] \quad , n \geq 1, C_{n+1} = C_n + W_{n+1}$$

$$\frac{\pi(A_{T-2}|S_{T-2})\ \pi(A_{T-1}|S_{T-1})}{b(A_{T-2}|S_{T-2})\ b(A_{T-1}|S_{T-1})}$$

**<u>Off policy Monte-Carlo prediction Algorithm:</u>**

**Incremental off-policy every-visit MC policy evaluation**

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
- $Q(s,a) \leftarrow$ arbitrary
- $C(s,a) \leftarrow 0$
- $\mu(a|s) \leftarrow$ an arbitrary soft behavior policy
- $\pi(a|s) \leftarrow$ an arbitrary target policy

Repeat forever:
- Generate an episode using $\mu$:
  - $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T$
- $G \leftarrow 0$
- $W \leftarrow 1$
- For $t = T-1, T-2, \ldots$ downto 0:
  - $G \leftarrow \gamma G + R_{t+1}$
  - $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
  - $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)}\left[G - Q(S_t, A_t)\right]$
  - $W \leftarrow W \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}$
  - If $W = 0$ then ExitForLoop

**<u>Off policy Monte-Carlo control Algorithm:</u>**

**Off-policy every-visit MC control (returns $\pi \approx \pi_*$)**

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
- $Q(s,a) \leftarrow$ arbitrary
- $C(s,a) \leftarrow 0$
- $\pi(s) \leftarrow$ a deterministic policy that is greedy with respect to $Q$

Repeat forever:
- Generate an episode using any soft policy $\mu$:
  - $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T$
- $G \leftarrow 0$
- $W \leftarrow 1$
- For $t = T-1, T-2, \ldots$ downto 0:

Generate an episode using any soft policy $\mu$:

$S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T$

$G \leftarrow 0$

$W \leftarrow 1$

For $t = T-1, T-2, \ldots$ downto 0:

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

$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$

$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$

$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$ (with ties broken consistently)

If $A_t \neq \pi(S_t)$ then ExitForLoop

$W \leftarrow W \frac{1}{\mu(A_t|S_t)}$