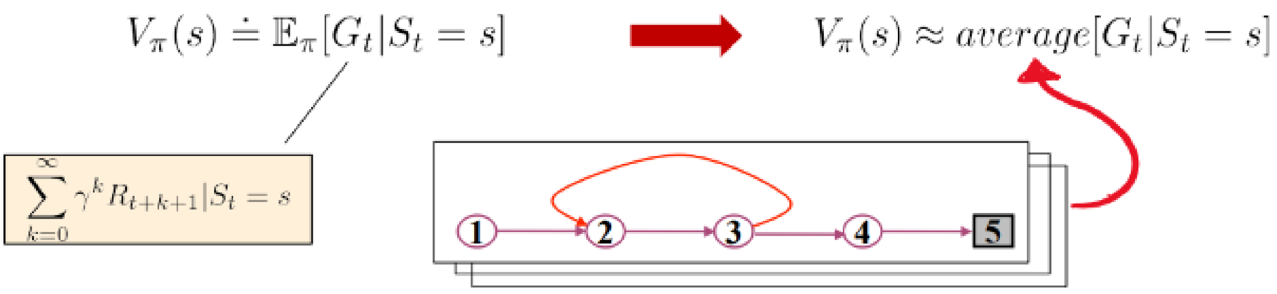
Montecarlo Methods

Solves the problem of not knowing one step dynamic of my problem.
Monte Carlo methods can be used in two ways:

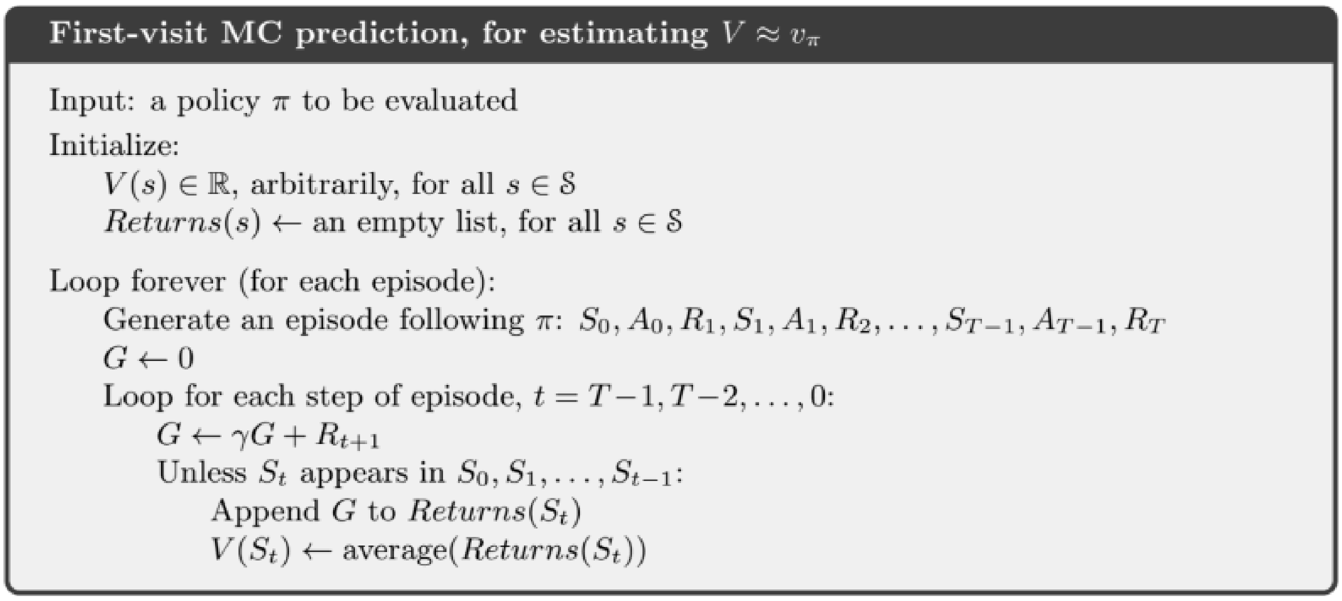
- model-free: no model necessary and still attains optimality
- simulated: needs only a simulation, not a full model
  Monte Carlo methods learn from complete sample returns
  Only defined for episodic tasks
  I need several path of solution but each of them need to be complete(cannot learn from partial solution of the Markov Decision Problem)

Monte Carlo Policy Evaluation

- Goal: learn $V_\pi(s)$
- Given: some number of episodes under π which contain s
- Idea: Average returns observed after visits to s


$V_\pi(s) \doteq \mathbb{E}_\pi[G_t|S_t = s]$
$V_\pi(s) \approx average[G_t|S_t = s]$
$\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s$
1
2
3
4
5

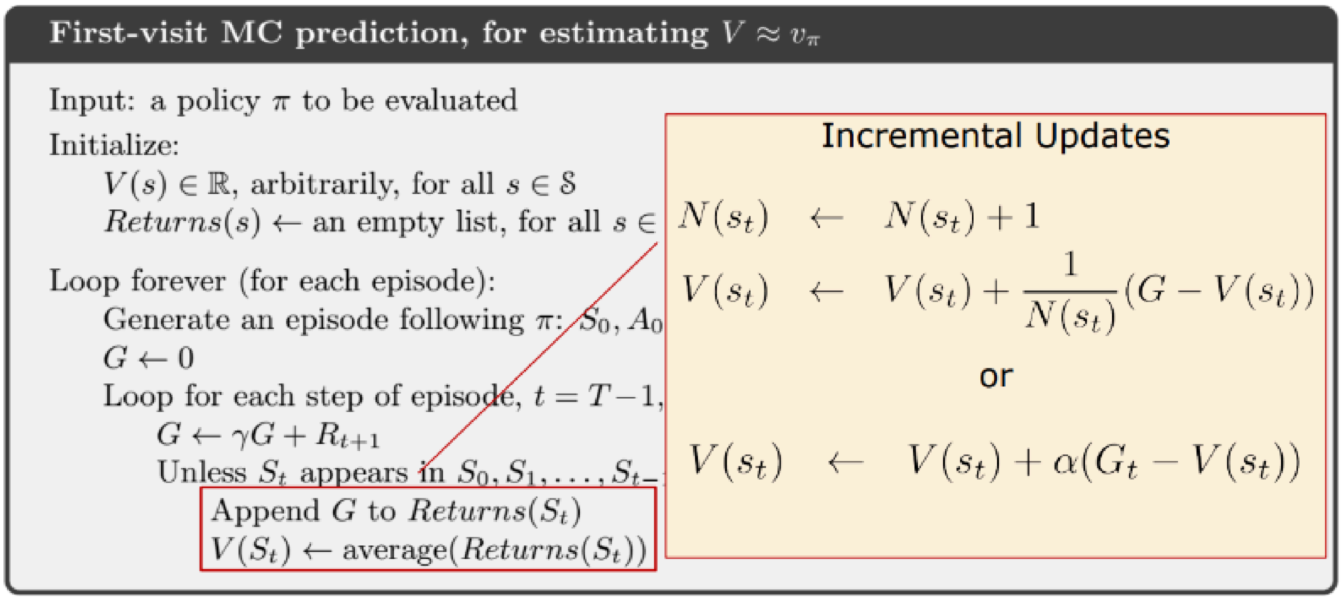
Every-Visit MC: average returns for every time s is visited in an episode
First-visit MC: average returns only for first time s is visited in an episode
Both converge asymptotically
We can observe the value returned several time at various observations. Just compute the value function as the average of G.



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, $t = T-1, T-2, \ldots, 0$:
        $G \leftarrow \gamma G + R_{t+1}$
        Unless $S_t$ appears in $S_0, S_1, \ldots, S_{t-1}$:
            Append $G$ to $Returns(S_t)$
            $V(S_t) \leftarrow \text{average}(Returns(S_t))$
```

To keep it easy we can use an incremental average:



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 $s \in$

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

Incremental Updates

$$N(s_t) \leftarrow N(s_t) + 1$$

$$V(s_t) \leftarrow V(s_t) + \frac{1}{N(s_t)}(G - V(s_t))$$

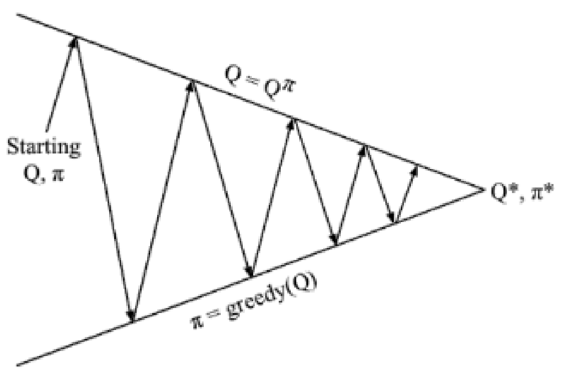
or

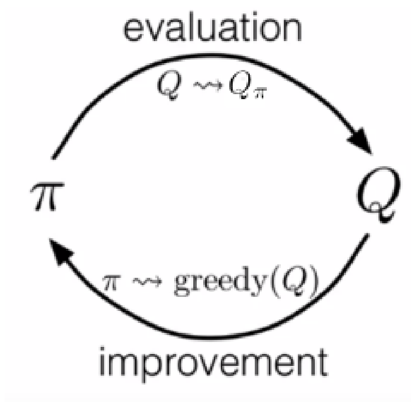
$$V(s_t) \leftarrow V(s_t) + \alpha(G_t - V(s_t))$$

If you expect the problem to change in time this will keep updates of the various values in time Montecarlo is useful when it is easy to create an infinite number of episodes to train your algorithm.

Policy Iteration

The idea is that the things are less straightforward than expected.
The idea is that the policy improvement is: $\pi'(s) = \underset{a}{argmax}\{r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V_\pi(s')\}$
It is impossible in Montecarlo so I apply the action by the function not requiring information




evaluation
$Q \rightsquigarrow Q_\pi$
π
Q
$\pi \rightsquigarrow \text{greedy}(Q)$
improvement

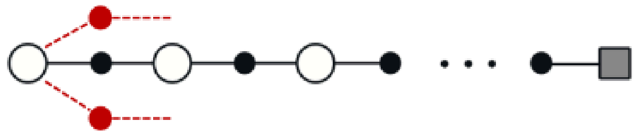
The price for doing that I need to compute the empirical average of the pair states(all possible sequence starting from state s and applying action a)

- ❑ We average return starting from state *s* and action *a* following π:

$$Q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \Longrightarrow Q_\pi(s,a) \approx average[G_t | S_t = s, A_t = a]$$

- ❑ Converges asymptotically **if every** state-action pair is visited:

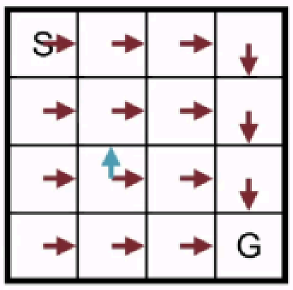
Exploration



- ❑ Exploring Starts

$s_0, a_0, s_1, a_1, s_2, a_2, \ldots$

Random From π and p

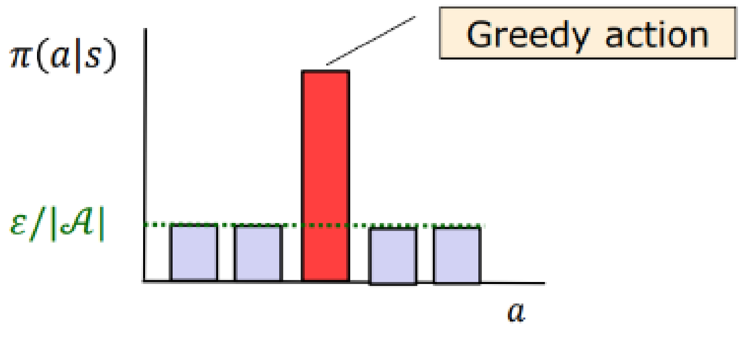


```
Monte Carlo ES (Exploring Starts), for estimating $\pi \approx \pi_*$

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 $Returns(S_t, A_t)$
            $Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$
            $\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$
```

ϵ-Soft Monte Carlo Policy Iteration

An ϵ-Soft Monte Carlo Policy is a policy that guarantee that all the action in the policy have a small probability to be chosen.

$$\pi(a|s) = \begin{cases} \frac{\epsilon}{|\mathcal{A}(s)|} + 1 - \epsilon & \text{if } a^* = arg\max_{a \in \mathcal{A}} Q(s,a) \\ \frac{\epsilon}{|\mathcal{A}(s)|} & \text{otherwise} \end{cases}$$



The larger the ϵ the larger the exploration
The ϵ greedy policy guarantee at least the probability and the rest of probability is dedicated to the best action in accordance to the model in use(greedy as it use almost all budget for the best action). The idea is that in order to guarantee some level of exploration we shift from an idea of learning a deterministic policy to learn a probabilistic policy. This guarantee some level of exploration. With ϵ =0 the policy is completely greedy.

```
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 $\pi$: $S_0, A_0, R_1, \dots, S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    Loop for each step of episode, $t = T-1, T-2, \dots, 0$:
        $G \leftarrow \gamma G + R_{t+1}$
        Unless the pair $S_t, A_t$ appears in $S_0, A_0, S_1, A_1 \dots, S_{t-1}, A_{t-1}$:
            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 $a \in \mathcal{A}(S_t)$:
                $\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$
```

By doing the policy improvement I will obtain a policy that is at least equal to my policy, usually it is better.

- **Thorem**
  Any ε-greedy policy π' with respect to Q_π is an improvement over any ε-soft policy π
- **Proof**

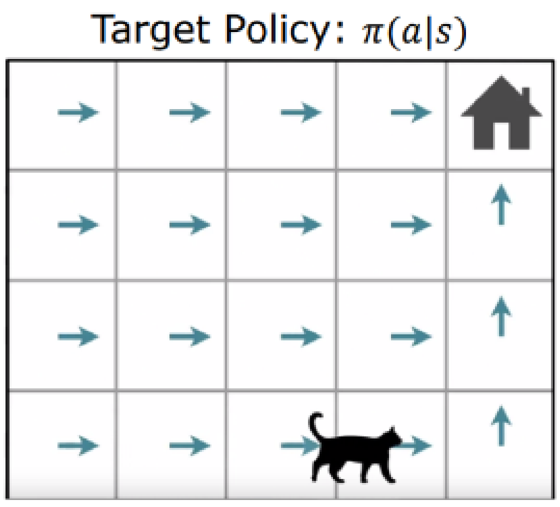
$$
\begin{aligned}
V_\pi(s) &= Q_\pi(s, \pi'(s)) \\
&= \sum_{a \in \mathcal{A}} \pi'(a|s) Q_\pi(s,a) \\
&= \frac{\epsilon}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} Q_\pi(s,a) + (1-\epsilon) \max_{a \in \mathcal{A}} Q^\pi(s,a) \\
&\geq \frac{\epsilon}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} Q_\pi(s,a) + (1-\epsilon) \sum_{a \in \mathcal{A}} \frac{\pi(a|s) - \frac{\epsilon}{|\mathcal{A}|}}{1-\epsilon} Q_\pi(s,a) \\
&= \sum_{a \in \mathcal{A}} \pi(a|s) Q_\pi(s,a) = V_\pi(s)
\end{aligned}
$$

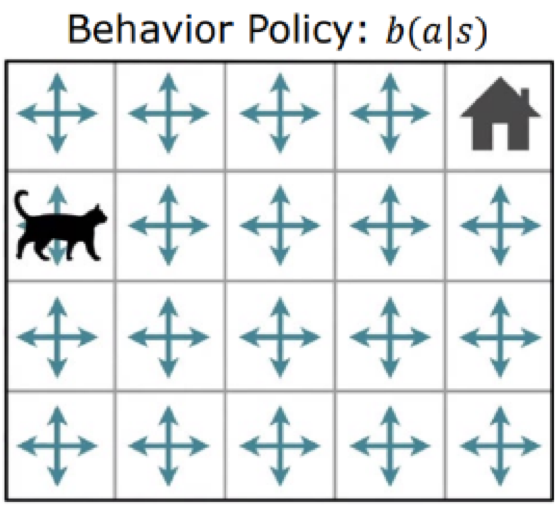
Off-policy Learning

On-Policy Learning

- The agent learns the value functions of the same policy used to select actions
- Exploration/Exploitation Dilemma: we cannot easily learn an optimal deterministic policy
  Off-Policy Learning
- The agent select action using a behaviour policy b(a|s), used to generate data

- These data is used to learn the value functions of a different target policy π(a|s), policy I want to evaluate and improve
- While b(a|s) can be an explorative policy, the agent can learn an optimal deterministic policy $\pi^*(a|s)$





We need to use the technique called **Importancce Sampling**. One of its limitations is that no matter the technique used is always impossible to learn a policy π that choose an action in the state s if this cannot happen in b. Importante sampling allow to estimate expectation of a distribution different w.r.t. the distribution used to draw the samples.

$$\boxed{\rho(x) = \frac{p(x)}{q(x)}}$$

$$\mathbb{E}_p[x] = \sum_{x \in X} x p(x) = \sum_{x \in X} x \frac{p(x)}{q(x)} q(x) = \sum_{x \in X} x \rho(x) q(x) = \mathbb{E}_q[x\rho(x)]$$

Accordingly, we can use this for sample-based estimate:

$$\mathbb{E}_p[x] \approx \frac{1}{N}\sum_{i=1}^{N} x_i \text{ if } x_i \sim p(x) \quad \Longrightarrow \quad \mathbb{E}_p[x] \approx \frac{1}{N}\sum_{i=1}^{N} x_i \rho(x_i) \text{ if } x_i \sim q(x)$$

When following policy π we computed the state value function as:

$$V_\pi(s) \approx average(Returns[0], Returns[1], Returns[2], \dots)$$

But when following policy b, the value function becomes:

$$V_\pi(s) \approx average(\rho_0 Returns[0], \rho_1 Returns[1], \rho_2 Returns[2], \dots)$$

Where ρ_i is the probability of performing the trajectory observed in episode i while following policy π over the probability of observing the same trajectory while following policy b

$$\rho = \frac{Prob[\text{trajectory under } \pi]}{Prob[\text{trajectory under } b]}$$

$$\begin{aligned}
&\Pr\{A_t, S_{t+1}, A_{t+1}, \dots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\
&\quad = \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1}) \cdots p(S_T|S_{T-1}, A_{T-1}) \\
&\quad = \prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k),
\end{aligned}$$

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

We also want to know the transition probability(we cannot in Monte Carlo) so we need to compute a fraction so using a new policy b the transition probability is up and down the fraction and so I only need to know the two policies.

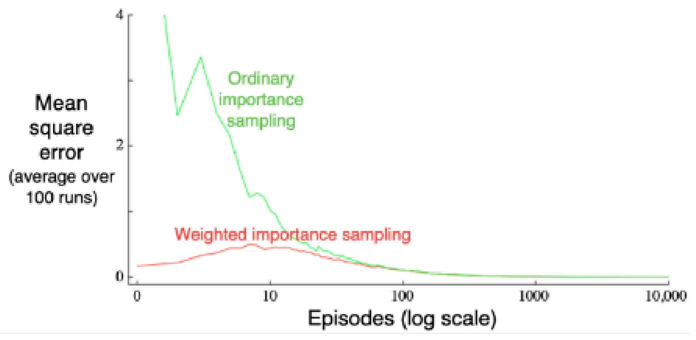
- Ordinary sampling

$$V_\pi(s) \approx \frac{\sum_i \rho[i] \cdot Return[i]}{N(s)}$$

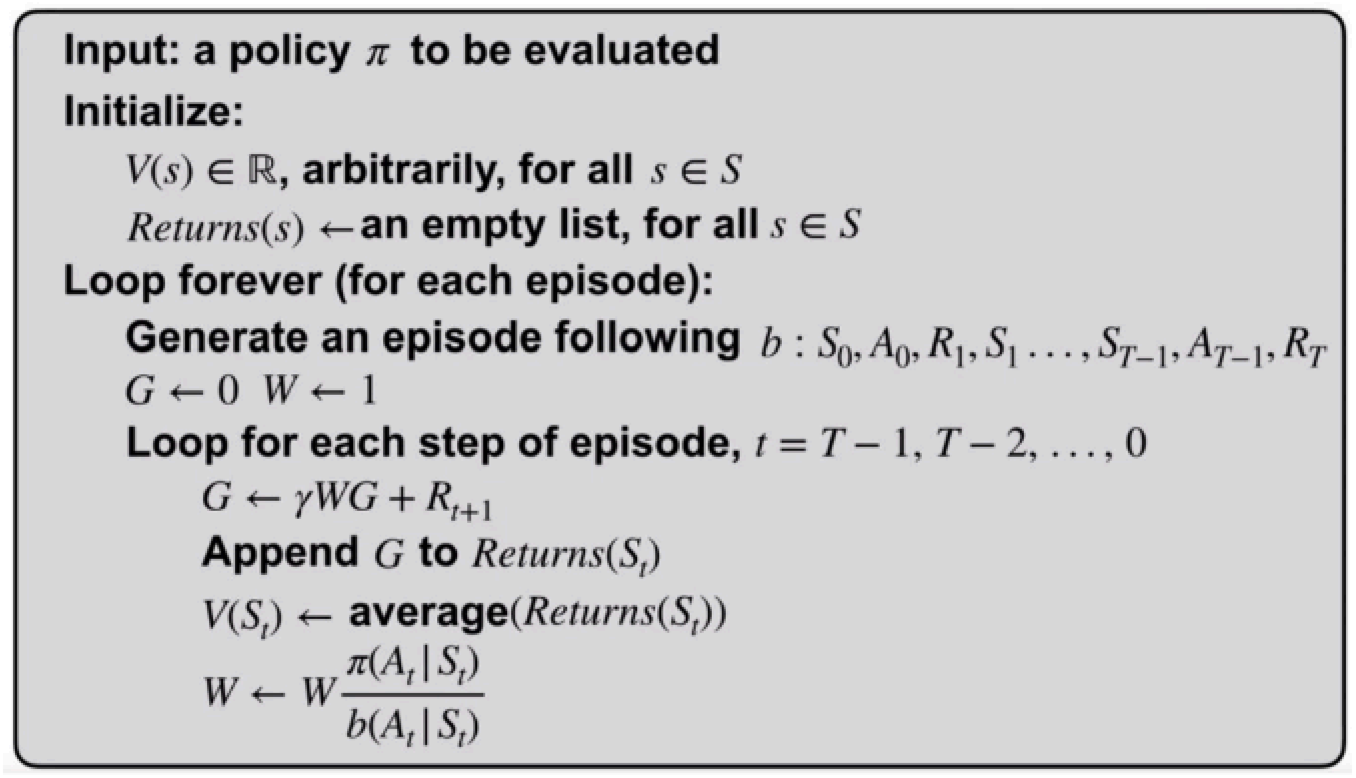
  - Unbiased
  - Higher Variance
- Weighted sampling

$$V_\pi(s) \approx \frac{\sum_i \rho[i] \cdot Return[i]}{\sum_i \rho[i]}$$

  - Biased (bias converges to zero)
  - Lower Variance



The correct algorithm is that you apply exactly the equation, so we observe all the return from state s and divide it for how many time I visited state s in my data(Unbiased estimator for the value function in s). An alternative solution is to compute a biased estimator that compute the sum of probability of all the trajectory of the state s. This estimator is biased and the bias will converge to zero, but on the other hand the estimator will have a low variance and suffer less from problem when starting with episodes with very few data.



```
Input: a policy $\pi$ to be evaluated
Initialize:
    $V(s) \in \mathbb{R}$, arbitrarily, for all $s \in S$
    $Returns(s) \leftarrow$ an empty list, for all $s \in S$
Loop forever (for each episode):
    Generate an episode following $b : S_0, A_0, R_1, S_1 \ldots, S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0 \;\; W \leftarrow 1$
    Loop for each step of episode, $t = T-1, T-2, \ldots, 0$
        $G \leftarrow \gamma W G + R_{t+1}$
        Append $G$ to $Returns(S_t)$
        $V(S_t) \leftarrow$ average$(Returns(S_t))$
        $W \leftarrow W \frac{\pi(A_t | S_t)}{b(A_t | S_t)}$
```

Completely decouple the policy that generate data from the policy that need to be evaluate.