

Metody Monte Carlo

Metody Monte Carlo

- Równania Bellmana można rozwiązać analitycznie.
- Złożoność obliczeniowa sięga $O(n^3)$ gdzie n oznacza liczbę stanów.
- W przypadku rozwiązywania dużych problemów istnieje wiele metod iteracyjnych:
 - Exhaustive search
 - Programowanie dynamiczne
 - Metody Monte Carlo
 - Temporal Difference Learning

Metody Monte Carlo

- Kluczowym problemem w przypadku korzystania z metod programowania dynamicznego jest fakt, że do ich rozwiązania konieczna jest znajomość pełnego modelu.

Metody Monte Carlo

- Kluczowym problemem w przypadku korzystania z metod programowania dynamicznego jest fakt, że do ich rozwiązania konieczna jest znajomość pełnego modelu.
- Oznacza to, że przed rozpoczęciem *planowania* musimy znać zarówno model przejścia $P(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ – jak i wszystkie osiągalne nagrody : $R(\mathbf{s}, \mathbf{a}, \mathbf{s}')$.

Metody Monte Carlo

- Kluczowym problemem w przypadku korzystania z metod programowania dynamicznego jest fakt, że do ich rozwiązania konieczna jest znajomość pełnego modelu.
- Oznacza to, że przed rozpoczęciem *planowania* musimy znać zarówno model przejścia $P(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ – jak i wszystkie osiągalne nagrody : $R(\mathbf{s}, \mathbf{a}, \mathbf{s}')$.
- Wykorzystując bezmodelowe metody uczenia jesteśmy w stanie obejść ten problem.

Metody Monte Carlo

- Kluczowym problemem w przypadku korzystania z metod programowania dynamicznego jest fakt, że do ich rozwiązania konieczna jest znajomość pełnego modelu.
- Oznacza to, że przed rozpoczęciem *planowania* musimy znać zarówno model przejścia $P(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ – jak i wszystkie osiągalne nagrody : $R(\mathbf{s}, \mathbf{a}, \mathbf{s}')$.
- Wykorzystując bezmodelowe metody uczenia jesteśmy w stanie obejść ten problem.
- W takim wypadku agent uczy się na podstawie swojego **doświadczenia**, poprzez powtarzające się **interakcje** z otoczeniem.

Metody Monte Carlo

- Podstawowym sposobem uczenia bez modelu są **Metody Monte Carlo**.
- Uczenie opiera się na generowaniu **pełnych epizodów** – agent nie bootstrapuje swojego doświadczenia na podstawie wcześniejszych wyników.

Metody Monte Carlo

- Podstawowym sposobem uczenia bez modelu są **Metody Monte Carlo**.
- Uczenie opiera się na generowaniu **pełnych epizodów** – agent nie bootstrapuje swojego doświadczenia na podstawie wcześniejszych wyników.
 - Oznacza to, że metody Monte Carlo można stosować jedynie w przypadku skończonych procesów decyzyjnych Markowa.

Metody Monte Carlo

- Podstawowym sposobem uczenia bez modelu są **Metody Monte Carlo**.
- Uczenie opiera się na generowaniu **pełnych epizodów** – agent nie bootstrapuje swojego doświadczenia na podstawie wcześniejszych wyników.
 - Oznacza to, że metody Monte Carlo można stosować jedynie w przypadku skończonych procesów decyzyjnych Markowa.
 - Ale też oznacza to, że możliwe jest uczenie się problemów, które nie spełniają własności Markowa.

Metody Monte Carlo

- Idea uczenia MC jest prosta:
 - Wartość funkcji wartości w k -tej iteracji jest równa przeciętnej skumulowanej przyszłej nagrodzie otrzymanej w poprzednich iteracjach.
 - Na mocy prawa wielkich liczb, gdy ilość symulowanych epizodów będzie dążyła do nieskończoności to oszacowanie będzie dążyło do prawdziwej funkcji wartości.

Monte Carlo Prediction

- **Ewaluacja** – dana jest strategia π , należy dokonać jej oceny:
 - Zainicjuj algorytm wybierając dowolne początkowe wartości funkcji wartości, np.
 $V_0(s) = 0 \forall s$.

W każdym kroku k :

- Wygeneruj epizod na podstawie strategii
 π : $s_0, a_0, r_1, \dots, s_{T-1}, a_{T-1}, r_T$
- Dla każdego ze stanów $s \in S$ należących do wygenerowanego epizodu, **gdym odwiedzasz go za pierwszym razem**:
 - Uaktualnij licznik odwiedzin stanu s :
$$n_s = n_s + 1$$
 - Uaktualnij przeciętną wartość funkcji wartości:
$$V_k(s) = V_{k-1}(s) + \frac{1}{n_s} (R_k(s) - V_{k-1}(s))$$

W każdym kroku k :

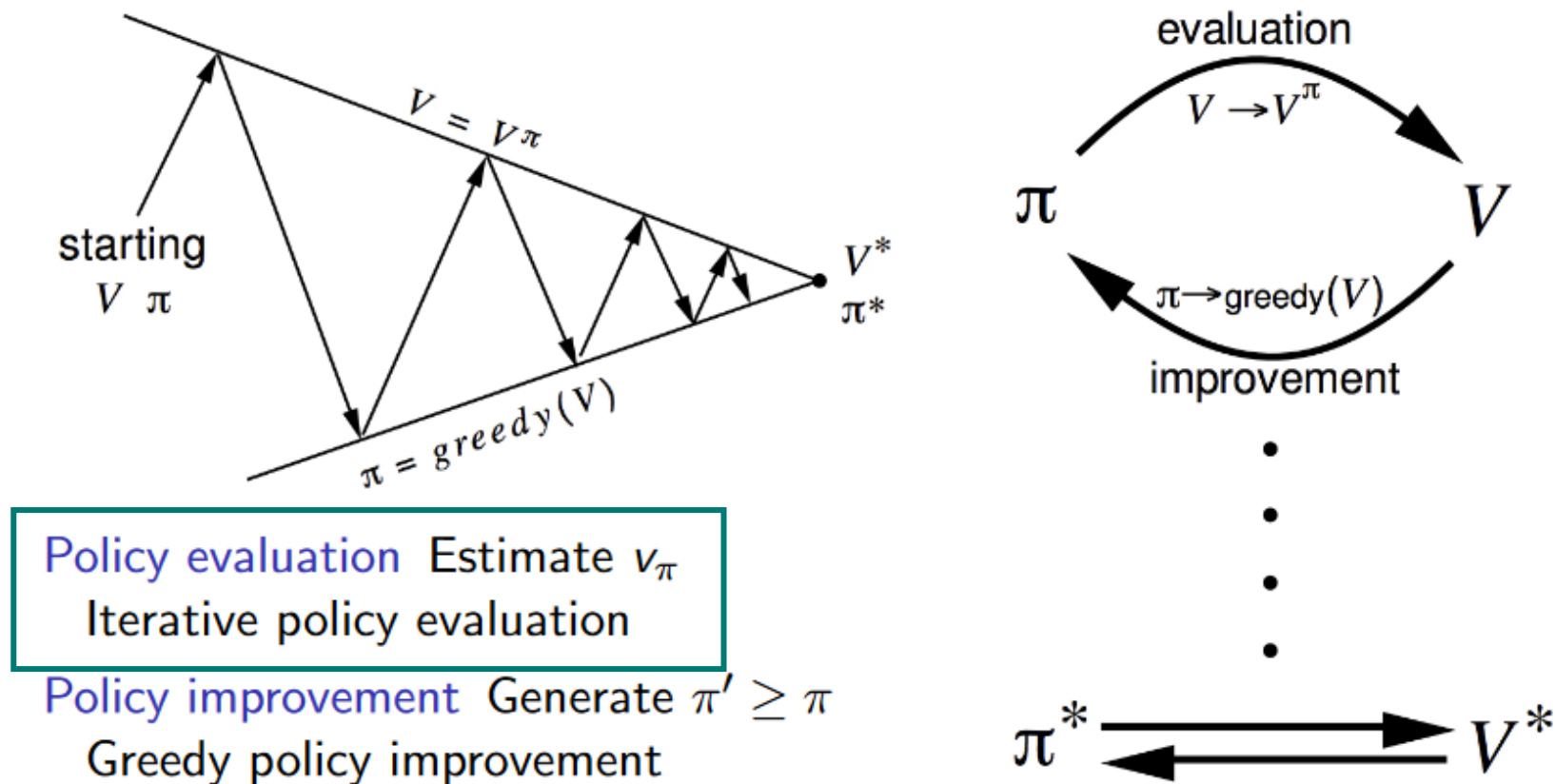
- Wygeneruj epizod na podstawie strategii
 π : $s_0, a_0, r_1, \dots, s_{T-1}, a_{T-1}, r_T$
- Dla każdego ze stanów $s \in S$ należących do wygenerowanego epizodu, **za każdym razem gdy go odwiedzasz**:
 - Uaktualnij licznik odwiedzin stanu s :
$$n_s = n_s + 1$$
 - Uaktualnij przeciętną wartość funkcji wartości:
$$V_k(s) = V_{k-1}(s) + \frac{1}{n_s} (R_k(s) - V_{k-1}(s))$$

- Obie metody zbiegną $V(s) \rightarrow V_\pi(s)$ gdy $n_s \rightarrow \infty$.

Monte Carlo Control

- W przypadku poszukiwania optymalnej strategii π_* za pomocą metod Monte Carlo konieczna jest modyfikacja podejścia do rozwiązywanego problemu:

Monte Carlo Control



Monte Carlo Control

- Zachłanna iteracja po funkcji $V(s)$ jest niemożliwa; wymaga znajomości modelu:

$$\pi' = \operatorname{argmax}_a \sum_{s,r} P(s, a, s')(r + V_{\pi}(s'))$$

Monte Carlo Control

- Zachłanna iteracja po funkcji $V(s)$ jest niemożliwa; wymaga znajomości modelu:

$$\pi' = \operatorname{argmax}_{a \in A} \sum_{s, r} P(s, a, s') (r + V_{\pi}(s'))$$

- Konieczne jest wykorzystanie funkcji $Q(s, a)$:

$$\pi' = \operatorname{argmax}_{a \in A} Q(s, a)$$

Monte Carlo Control z Eksploracją punktów startowych

1. Zainicjuj algorytm wybierając dowolne początkowe wartości funkcji wartości, np. $Q_0(s, a) = 0 \forall s, a$, wektor odwiedzin part stan i akcja: $N(s, a) = 0 \forall s, a$ i dowolną strategię π .
2. W każdej iteracji k :
 - Wylosuj początkową parę $s_0 \in S, a_0 \in A$
 - Wygeneruj epizod na podstawie strategii π : $s_0, a_0, r_1, \dots, s_{T-1}, a_{T-1}, r_T$.
 - Przyjmij wartość skumulowanej przyszłej nagrody $R = 0$
 - Dla każdego $t = T - 1, T - 2, \dots, 1, 0$:
 - Przypisz nową wartość $R = \beta R + r_{t+1}$
 - Jeżeli para s_t, a_t nie występuje w $s_0, a_0, r_1, \dots, s_{t-1}, a_{t-1}$ (**jest pierwszym wystąpieniem**):
 - Uaktualnij licznik odwiedzin pary s_t, a_t :
$$N(s_t, a_t) = N(s_t, a_t) + 1$$
 - Uaktualnij funkcję wartości akcji:
$$Q(s_t, a_t) = Q(s_t, a_t) + \frac{1}{N(s, a)} (R - Q(s_t, a_t))$$
 - Popraw zaproponowaną strategię zachowując się zachłannie:
$$\pi(s) = \operatorname{argmax}_{a \in A} Q(s, a)$$
3. Gdy $n \rightarrow \infty$ to $\pi \approx \pi_*$

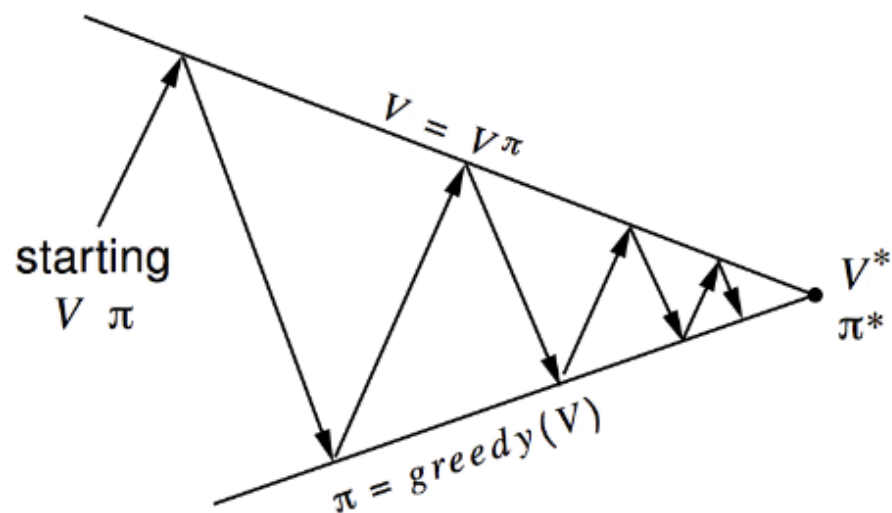
Przykład

Frozen Lake cd.

S	F	F	F
F	H	F	H
F	F	F	H
H	F	F	G

- Celem agenta jest przejść z punktu **S** do Punktu **G**.
- Agent może iść po lodzie (pola oznaczone literą **F**), musi unikać wpadnięcia do przerębli (pola oznaczone jako **H**).
- Lód jest śliski; idąc przed siebie z pewnym prawdopodobieństwem p może się poślizgnąć i przesunąć w lewo lub w prawo w stosunku do swojej wyjściowej pozycji.

Monte Carlo Control

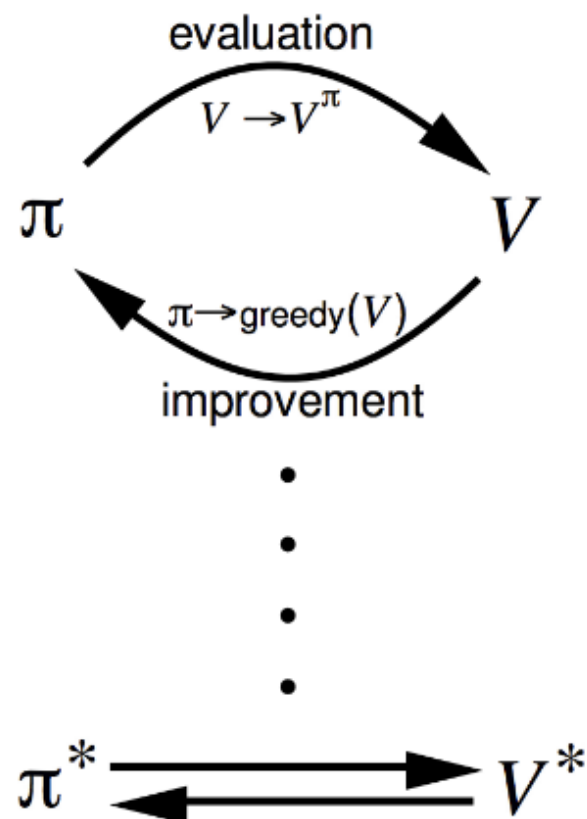


Policy evaluation Estimate v_π

Iterative policy evaluation

Policy improvement Generate $\pi' \geq \pi$

Greedy policy improvement



Monte Carlo Control

- Problematyczny jest też fakt zapewnienia odpowiedniej ilości epizodów na podstawie których możliwe będzie oszacowanie $Q(s, a)$.
 - Eksploracja
 - Eksploatacja

Monte Carlo Control

- Problematyczny jest też fakt zapewnienia odpowiedniej ilości epizodów na podstawie których możliwe będzie oszacowanie $Q(s, a)$.
 - Eksploracja
 - Eksploatacja
- Tradycyjny algorytm zachłanny bardzo szybko zacznie eksploatować rozwiązanie suboptymalne.

Monte Carlo Control

- **Strategia ϵ -zachłanna (ϵ -greedy)**

- Najprostszy algorytm pozwalający na zachowanie ciągłej eksploatacji w trakcie uczenia się.
- Zapewnia, że każda z dopuszczalnych m akcji będzie wybierana z niezerowym prawdopodobieństwem.
 - Z $p = 1 - \epsilon$ wybierz akcję zachłannie.
 - W przeciwnym wypadku wybieraj losowo.

Monte Carlo Control

- **Strategia ϵ -zachłanna (ϵ -greedy)**

$$\pi(a|s) = \begin{cases} \frac{\epsilon}{m} + 1 - \epsilon, & \text{gdy } a_* = \operatorname{argmax}_{a \in A} Q(s, a) \\ \frac{\epsilon}{m}, & \text{w przeciwnym wypadku} \end{cases}$$

Monte Carlo Control

- **On-policy**

- Zoptymalizuj strategię π bazując na doświadczeniu próbkowanym na podstawie tej właśnie strategii. („*ucz się na swoich błędach*”)

- **Off-policy**

- Zoptymalizuj strategię π bazując na doświadczeniu próbkowanym z innych strategii μ . („*ucz się na czyichś błędach*”)

Monte Carlo On Policy

1. Zainicjuj algorytm wybierając dowolne początkowe wartości funkcji wartości, np. $Q_0(s, a) = 0 \forall s, a$, wektor odwiedzin part stan i akcja: $N(s, a) = 0 \forall s, a$ i dowolną ϵ -zachłanną strategię π .
2. W każdej iteracji k :
 - Wygeneruj epizod na podstawie strategii π : $s_0, a_0, r_1, \dots, s_{T-1}, a_{T-1}, r_T$.
 - Przyjmij wartość skumulowanej przyszłej nagrody $R = 0$
 - Dla każdego $t = T - 1, T - 2, \dots, 1, 0$:
 - Przypisz nową wartość $R = \beta R + r_{t+1}$
 - Jeżeli para s_t, a_t nie występuje w $s_0, a_0, r_1, \dots, s_{t-1}, a_{t-1}$ (**jest pierwszym wystąpieniem**):
 - Uaktualnij licznik odwiedzin pary s_t, a_t :
$$N(s_t, a_t) = N(s_t, a_t) + 1$$
 - Uaktualnij funkcję wartości akcji:
$$Q(s_t, a_t) = Q(s_t, a_t) + \frac{1}{N(s, a)} (R - Q(s_t, a_t))$$
 - Popraw zaproponowaną strategię zachowując się zachłannie:
$$\pi(a|s) \leftarrow \begin{cases} 1 - \epsilon + \epsilon / |A(S_t)| & \text{gdy } a = \underset{a \in A}{\operatorname{argmax}} Q(s, a) \\ \epsilon / |A(S_t)| & \text{gdy } a \neq \underset{a \in A}{\operatorname{argmax}} Q(s, a) \end{cases} \quad \forall a \in A(S_t)$$
3. Gdy $n \rightarrow \infty$ to $\pi \approx \pi_*$

Przykład

Frozen Lake cd.

S	F	F	F
F	H	F	H
F	F	F	H
H	F	F	G

- Celem agenta jest przejść z punktu **S** do Punktu **G**.
- Agent może iść po lodzie (pola oznaczone literą **F**), musi unikać wpadnięcia do przerębli (pola oznaczone jako **H**).
- Lód jest śliski; idąc przed siebie z pewnym prawdopodobieństwem p może się poślizgnąć i przesunąć w lewo lub w prawo w stosunku do swojej wyjściowej pozycji.

Eksploracja i eksploatacja

- On-policy
 - Zoptymalizuj strategię π bazując na doświadczeniu próbkowanym na podstawie tej właśnie strategii. („*ucz się na swoich błędach*”)
- Off-policy
 - Zoptymalizuj strategię π bazując na doświadczeniu próbkowanym z innych strategii μ . („*ucz się na czyichś błędach*”)

Monte Carlo Off Policy

- Jednym z problemów związanych z uczeniem bezmodelowym jest to, że agent musi nauczyć zachowywać się **optymalnie** zgodnie z pewną optymalną strategią a zarazem eksplorować środowisko.
- W jaki sposób połączyć te dwa cele?

Monte Carlo Off Policy

- W przypadku uczenia on policy konieczny był pewien kompromis, zamiast zachowywać się zgodnie z zadaną deterministyczną strategią π agent korzystał z ϵ -zachłannej strategii π' , takiej że $\pi'(a|s) > 0 \forall A(s)$.

Monte Carlo Off Policy

- W przypadku uczenia on policy konieczny był pewien kompromis, zamiast zachowywać się zgodnie z zadaną deterministyczną strategią π agent korzystał z ϵ -zachłannej strategii π' , takiej że $\pi'(a|s) > 0 \forall A(s)$.
- Uczenie off policy jest dużo prostsze. Agent uczy się **docelowej strategii (*target policy*)** π za pomocą **strategii kontrolującej zachowanie (*behavior policy*)** μ takiej że
$$\pi(a|s) > 0 \rightarrow \mu(a|s) > 0 \forall A(s)$$
- Zakładamy, że obie strategię pokrywają taką samą przestrzeń (***assumption of coverage***).

Monte Carlo Off Policy

- Problemem jest to, że w ramach uczenia off policy chcemy nauczyć się jednej strategii za pomocą próbek generowanych z innej. Aby było to możliwe musimy wykorzystać technikę znaną jako **próbkiwanie** *ważności* (*importance sampling*).

Importance sampling

- Prawdopodobieństwo wystąpienia trajektorii $s_\tau, a_\tau, r_{\tau+1}, \dots, s_{T-1}, a_{T-1}, r_T$ w trakcie korzystania ze strategii π :

$$\Pr\{s_\tau, a_\tau, r_{\tau+1}, \dots, s_{T-1}, a_{T-1}, r_T | s_t, a_t \sim \pi\} = \prod_{t=\tau}^T \pi(a_t | s_t) P(s_t, a_t, s_{t+1})$$

- Relatywne prawdopodobieństwo wystąpienia trajektorii zarówno w strategii celu π jak i strategii zachowania μ (*importance sampling ratio*):

$$\rho_{\tau:T-1} = \frac{\prod_{t=\tau}^T \pi(a_t | s_t) P(s_t, a_t, s_{t+1})}{\prod_{t=\tau}^T \mu(a_t | s_t) P(s_t, a_t, s_{t+1})}$$

Importance sampling

- **Ordinary importance sampling**

- Przeskaluj wyniki korzystając z ρ i następnie uśrednij rezultat:

$$V(s) = \frac{\sum_{n \in N(s)} \rho_n R_n}{N(s)}$$

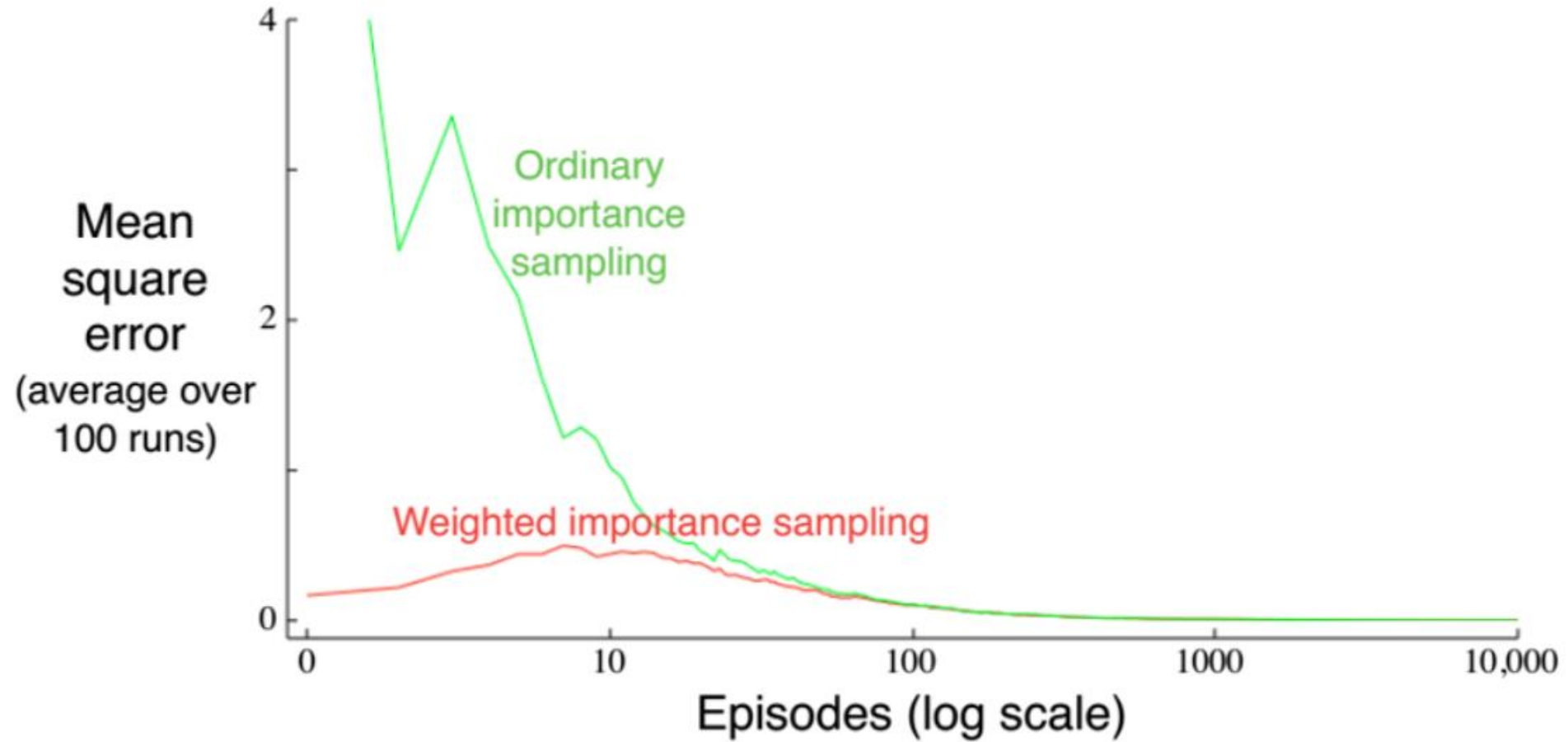
- **Weighted importance sampling**

- Wyznacz średnią ważoną korzystając z ρ :

$$V(s) = \frac{\sum_{n \in N(s)} \rho_n R_n}{\sum_{n \in N(s)} \rho_n}$$

- Gdzie $N(s)$ to licznik odwiedzin stanu.

Importance sampling



Źródło: Richard S. Sutton and Andrew G. Barto, *Reinforcement learning, an introduction*, second edition

Monte Carlo Off Policy

1. Zainicjuj algorytm wybierając dowolne początkowe wartości funkcji wartości, np. $Q_0(s, a) = 0 \forall s, a$, wektor odwiedzin part stan i akcja: $N(s, a) = 0 \forall s, a$ i dowolną strategię π .
2. W każdej iteracji k :
 - Wygeneruj epizod na podstawie strategii μ : $s_0, a_0, r_1, \dots, s_{T-1}, a_{T-1}, r_T$.
 - Przyjmij wartość skumulowanej przyszłej nagrody $R = 0$
 - Przyjmij wagę $W = 1$
 - Dla każdego $t = T - 1, T - 2, \dots, 1, 0$ gdy $W \neq 0$:
 - Przypisz nową wartość $R = \beta R + r_{t+1}$
 - Jeżeli para s_t, a_t nie występuje w $s_0, a_0, r_1, \dots, s_{t-1}, a_{t-1}$ (**jest pierwszym wystąpieniem**):
 - Uaktualnij licznik odwiedzin pary s_t, a_t :
$$N(s_t, a_t) = N(s_t, a_t) + W$$
 - Uaktualnij funkcję wartości akcji:
$$Q(s_t, a_t) = Q(s_t, a_t) + \frac{W}{N(s, a)} (R - Q(s_t, a_t))$$
 - Popraw zaproponowaną strategię zachowując się zachłannie:
$$\pi(a|s) \leftarrow \operatorname{argmax}_{a \in A} Q(s, a)$$
 - Jeżeli $a_t \neq \pi(s_t)$ przerwij pętlę (przejdź do kolejnego epizodu).
 - $W = \frac{W}{\mu(a_t|s_t)}$
3. Gdy $n \rightarrow \infty$ to $\pi \approx \pi_*$

Przykład

Blackjack

