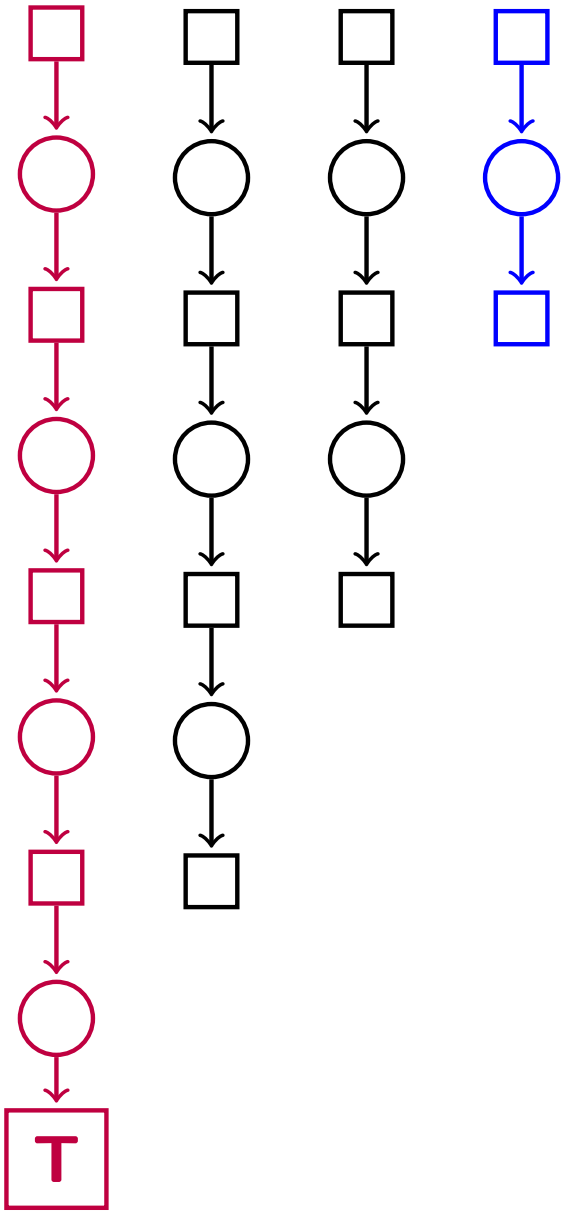
Medzi TD a MC
(Strojové učenie II)

M. Mach

Katedra kybernetiky a umelej inteligencie, FEI, TUKE

marec 2023

Odhad kumulatívnej odmeny



- Úplný odhad G (MC)
  $G_t = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-t-1} R_T$
- Jednokrokový odhad G (TD)
  $G_{t:t+1} = R_{t+1} + \gamma v_t(S_{t+1})$
- Dvojkrokový odhad G
  $G_{t:t+2} = R_{t+1} + \gamma R_{t+2} + \gamma^2 v_{t+1}(S_{t+2})$
- ...
- n-krokový odhad G
  $G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n v_{t+n-1}(S_{t+n})$
  ak $t + n \geq T$
  $G_{t:t+n} = G_t$

Aktualizácia hodnotovej funkcie

- Ilustrácia pre $n = 3$

$$S_1, S_2, S_3, S_4, \ldots, S_{T-3}, S_{T-2}, S_{T-1}, S_T, T+1, T+2$$

- V každom kroku sa aktualizuje hodnota jedného stavu
  - okrem prvých $n-1$ krokov začiatku epizódy
  - učí sa aj $n-1$ krokov po dosiahnutí terminálneho stavu
- Aktualizácia hodnotovej funkcie

$$v_{t+n}(S_t) = v_{t+n-1}(S_t) + \alpha\delta_{t:t+n}$$
$$v_{t+n}(s) = v_{t+n-1}(s) \text{ pre všetky } s \neq S_t$$

kde chyba $\delta_{t:t+n}$

$$\delta_{t:t+n} = [G_{t:t+n} - v_{t+n-1}(S_t)]$$

Algoritmus TD(n) odhadu v_π

n-step TD for estimating $V \approx v_\pi$

Input: a policy π
Algorithm parameters: step size $\alpha \in (0, 1]$, a positive integer n
Initialize $V(s)$ arbitrarily, for all $s \in \mathcal{S}$
All store and access operations (for S_t and R_t) can take their index mod $n+1$

Loop for each episode:
  Initialize and store $S_0 \neq$ terminal
  $T \leftarrow \infty$
  Loop for $t = 0, 1, 2, \ldots$:
  | If $t < T$, then:
  | Take an action according to $\pi(\cdot|S_t)$
  | Observe and store the next reward as R_{t+1} and the next state as S_{t+1}
  | If S_{t+1} is terminal, then $T \leftarrow t+1$
  | $\tau \leftarrow t - n + 1$ (τ is the time whose state's estimate is being updated)
  | If $\tau \geq 0$:
  | $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
  | If $\tau + n < T$, then: $G \leftarrow G + \gamma^n V(S_{\tau+n})$ $(G_{\tau:\tau+n})$
  | $V(S_\tau) \leftarrow V(S_\tau) + \alpha\,[G - V(S_\tau)]$
  Until $\tau = T - 1$

©*Sutton-Barto: Reinforcement Learning, 2nd ed., 2018*

Od vyhodnotenia politiky k jej učeniu

- Princíp
  - náhrada hodnotových funkcií $v_\pi(s) \rightarrow q_\pi(s,a)$
  - použitie ϵ-greedy politiky

$$
\begin{aligned}
G_{t:t+n} &= R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} \\
&\quad + \gamma^n q_{t+n-1}(S_{t+n}, A_{t+n}) && 0 \leq t < T - n \\
G_{t:t+n} &= G_t && t + n \geq T
\end{aligned}
$$

$$
\begin{aligned}
G_{t:t+n} &= R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} \\
&\quad + \gamma^n \textstyle\sum_a \pi(a|S_{t+n}) q_{t+n-1}(S_{t+n}, a) && 0 \leq t < T - n \\
G_{t:t+n} &= G_t && t + n \geq T
\end{aligned}
$$

$$
q_{t+n}(S_t, A_t) = q_{t+n-1}(S_t, A_t) + \alpha[G_{t:t+n} - q_{t+n-1}(S_t, A_t)]
$$

Algoritmus Sarsa(n) odhadu q

n-step Sarsa for estimating $Q \approx q_*$ or q_π

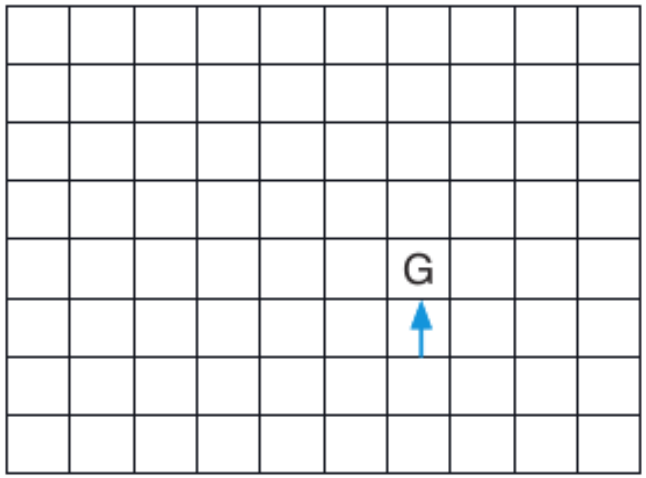
Initialize $Q(s,a)$ arbitrarily, for all $s \in \mathcal{S}, a \in \mathcal{A}$
Initialize π to be ε-greedy with respect to Q, or to a fixed given policy
Algorithm parameters: step size $\alpha \in (0,1]$, small $\varepsilon > 0$, a positive integer n
All store and access operations (for S_t, A_t, and R_t) can take their index mod $n+1$

Loop for each episode:
  Initialize and store $S_0 \neq$ terminal
  Select and store an action $A_0 \sim \pi(\cdot|S_0)$
  $T \leftarrow \infty$
  Loop for $t = 0, 1, 2, \ldots$:
  | If $t < T$, then:
  | | Take action A_t
  | | Observe and store the next reward as R_{t+1} and the next state as S_{t+1}
  | | If S_{t+1} is terminal, then:
  | | | $T \leftarrow t+1$
  | | else:
  | | | Select and store an action $A_{t+1} \sim \pi(\cdot|S_{t+1})$
  | $\tau \leftarrow t - n + 1$ (τ is the time whose estimate is being updated)
  | If $\tau \geq 0$:
  | | $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
  | | If $\tau + n < T$, then $G \leftarrow G + \gamma^n Q(S_{\tau+n}, A_{\tau+n})$ $(G_{\tau:\tau+n})$
  | | $Q(S_\tau, A_\tau) \leftarrow Q(S_\tau, A_\tau) + \alpha\left[G - Q(S_\tau, A_\tau)\right]$
  | | If π is being learned, then ensure that $\pi(\cdot|S_\tau)$ is ε-greedy wrt Q
  Until $\tau = T - 1$

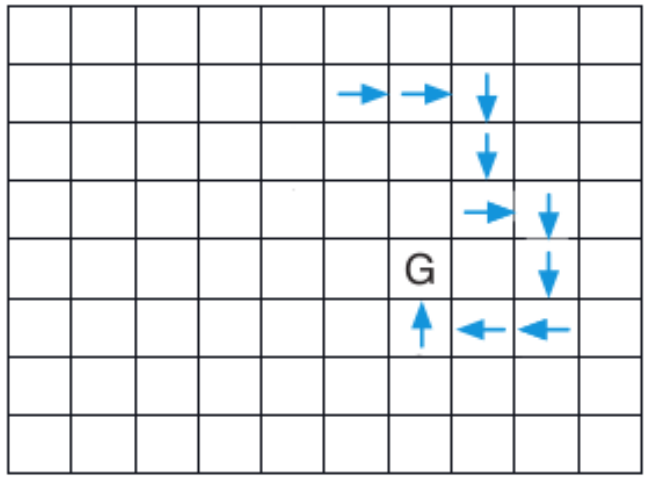
©*Sutton-Barto: Reinforcement Learning, 2nd ed., 2018*

Sarsa vs Sarsa(n)

Action values increased by one-step Sarsa



Action values increased by 10-step Sarsa



©*Sutton-Barto: Reinforcement Learning, 2nd ed., 2018*

Kombinovanie odmien

$$
\begin{aligned}
G_{t:t+n} &= R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} \\
&\quad + \gamma^n \hat{v}(S_{t+n}, w_{t+n-1}) && 0 \leq t < T - n \\
G_{t:t+n} &= G_t && t + n \geq T
\end{aligned}
$$

- V predchádzajúcom sa použil vždy iba jeden odhad kumulatívnej odmeny (pre nejaké konkrétne n)
- Kombinovanie viacerých odhadov do zloženého odhadu
  - kombinovanie váženým súčtom, súčet váh rovný 1
  - zložený update sa vykoná až potom, čo bude známy odhad s najväčším použitým n

λ-odmena

- λ-odmena používa kombinovanie odhadov podľa
$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_{t:t+n}$$
váhy: $(1-\lambda)$, $(1-\lambda)\lambda$, $(1-\lambda)\lambda^2$, ...
- V prípade epizódy s dĺžkou T

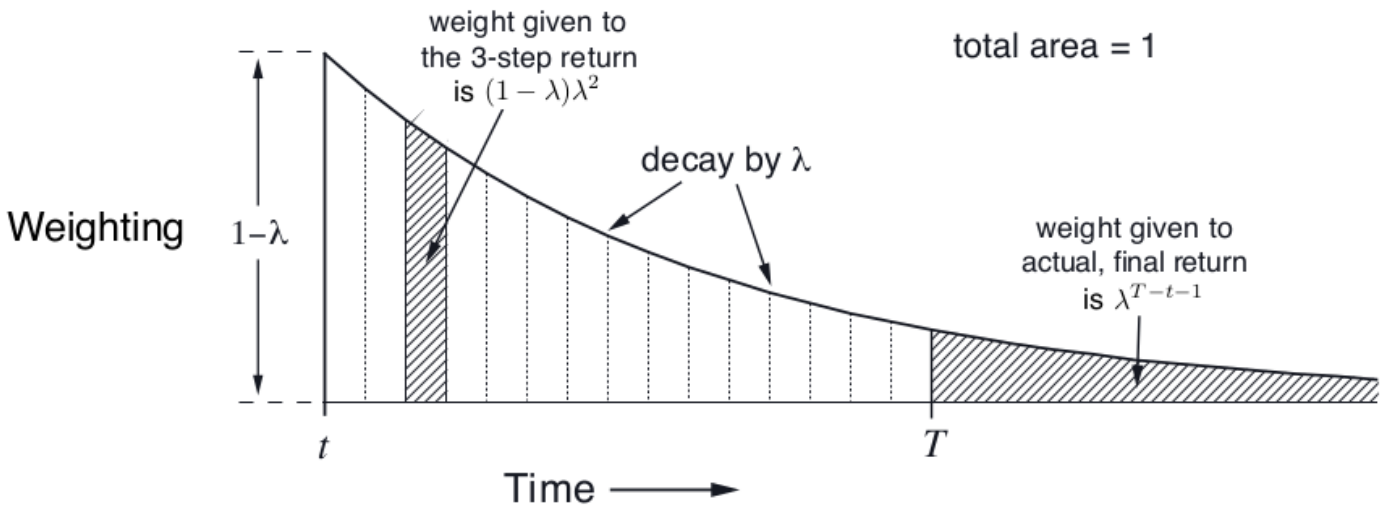
$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1} \lambda^{n-1} G_{t:t+n} + \lambda^{T-t-1} G_t$$

  - $\lambda = 0 \rightarrow G_t^\lambda = G_{t:t+1}$ (TD)
  - $\lambda = 1 \rightarrow G_t^\lambda = G_t$ (MC)
- Použitie v offline algoritme
  - offline - aktualizácie robí až po skončení epizódy
  - aktualizácia semi-gradientovým pravidlom

$$\bar{w}_{t+1} = \bar{w}_t + \alpha[G_t^\lambda - \hat{v}(S_t, \bar{w}_t)]\nabla \hat{v}(S_t, \bar{w}_t)$$

λ-odmena - odvodenie

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1}\lambda^{n-1}G_{t:t+n} + (1-\lambda)\sum_{n=T-t}^{\infty}\lambda^{n-1}G_{t:t+n}$$



©*Sutton-Barto: Reinforcement Learning, 2nd ed., 2018*

$$(1-\lambda)\sum_{n=T-t}^{\infty}\lambda^{n-1}G_{t:t+n} = (1-\lambda)\left(\sum_{n=T-t}^{\infty}\lambda^{n-1}\right)G_t$$
$$= (1-\lambda)\left(\sum_{n=1}^{\infty}\lambda^{T-t-1+n-1}\right)G_t = (1-\lambda)\left(\sum_{n=1}^{\infty}\lambda^{T-t-1}\lambda^{n-1}\right)G_t$$
$$= (1-\lambda)\left(\sum_{n=1}^{\infty}\lambda^{n-1}\right)\lambda^{T-t-1}G_t = (1-\lambda)\frac{1}{1-\lambda}\lambda^{T-t-1}G_t$$
$$= \lambda^{T-t-1}G_t$$

Dopredný a spätný pohľad

- Dopredný pohľad
  - pre každý navštívený stav sa uvažujú budúce odmeny a ich zloženie
  - budúce stavy sú používané opakovane (pre každý z predchádzajúcich stavov)
- Spätný pohľad
  - pre každý navštívený stav sa určí iba prvá budúca odmena
  - pre update sa uvažujú aj minulé stavy
  - budúce stavy nie sú používané opakovane

Eligibility traces

- Stopa spôsobilosti (ET) je $\bar{z}$, pričom $\mid \bar{z} \mid = \mid \bar{w} \mid$
  - ak $\bar{w}$ je dlhodobá pamäť (počas celého učenia), tak $\bar{z}$ je krátkodobá pamäť (iba v rámci epizódy)

$$\bar{z}_{-1} = 0$$
$$\bar{z}_t = \gamma\lambda\bar{z}_{t-1} + \nabla\hat{v}(S_t, \bar{w}_t) \qquad 0 \leq t \leq T$$

- Interpretácia ET
  - pamätá si, ktorá váha vektora $\bar{w}$ prispela pozitívne alebo negatívne
  - schopnosť zabúdať minulosť - minulé príspevky pomaly "blednú" faktorom $\gamma\lambda$
- Ovplyvňuje mieru aktualizácie jednotlivých zložiek $\bar{w}$

$$\delta_t = R_{t+1} + \gamma\hat{v}(S_{t+1}, \bar{w}_t) - \hat{v}(S_t, \bar{w}_t)$$
$$\bar{w}_{t+1} = \bar{w}_t + \alpha\delta_t\bar{z}_t$$

Algoritmus TD(λ) odhadu v_π

Semi-gradient TD(λ) for estimating $\hat{v} \approx v_\pi$

Input: the policy π to be evaluated
Input: a differentiable function $\hat{v} : \mathcal{S}^+ \times \mathbb{R}^d \to \mathbb{R}$ such that $\hat{v}(\text{terminal},\cdot) = 0$
Algorithm parameters: step size $\alpha > 0$, trace decay rate $\lambda \in [0, 1]$
Initialize value-function weights $\mathbf{w}$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Loop for each episode:
  Initialize S
  $\mathbf{z} \leftarrow \mathbf{0}$ (a d-dimensional vector)
  Loop for each step of episode:
  | Choose $A \sim \pi(\cdot|S)$
  | Take action A, observe R, S'
  | $\mathbf{z} \leftarrow \gamma\lambda\mathbf{z} + \nabla\hat{v}(S,\mathbf{w})$
  | $\delta \leftarrow R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$
  | $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
  | $S \leftarrow S'$
  until S' is terminal

©*Sutton-Barto: Reinforcement Learning, 2nd ed., 2018*

Analýza algoritmu TD(λ)

- $\lambda = 0$ (algoritmus TD(0))
  - ET je gradient korešpondujúci so stavom S_t
  - redukcia na jednokrokový algoritmus
- $0 < \lambda < 1$
  - čím je hodnota väčšia, tým viac predchádzajúcich stavov je zohľadňovaných
- $\lambda = 1$ (algoritmus TD(1))
  - minulé stavy sú diskontované pomocou γ
  - epizodické stavy nemusia byť diskontované ($\gamma = 1$)
  - redukcia na MC (zovšeobecnený MC so širším použitím)
    - môže byť aplikovaný aj na kontinuálny diskontovaný proces
    - je inkrementálny (nemusí čakať do konca epizódy) - vie reagovať okamžite na novú situáciu

Zmeny pre prácu s $\hat{q}$ namiesto $\hat{v}$

- Zmeny oproti TD(λ)
  - kumulatívna odmena

$$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n \hat{q}(S_{t+n}, A_{t+n}, w_{t+n-1}) \qquad t + n < T$$

  - stopa spôsobilosti

$$\bar{z}_{-1} = 0$$
$$\bar{z}_t = \gamma\lambda \bar{z}_{t-1} + \nabla \hat{q}(S_t, A_t, \bar{w}_t) \qquad 0 \leq t \leq T$$

  - aktualizácia $\bar{w}$

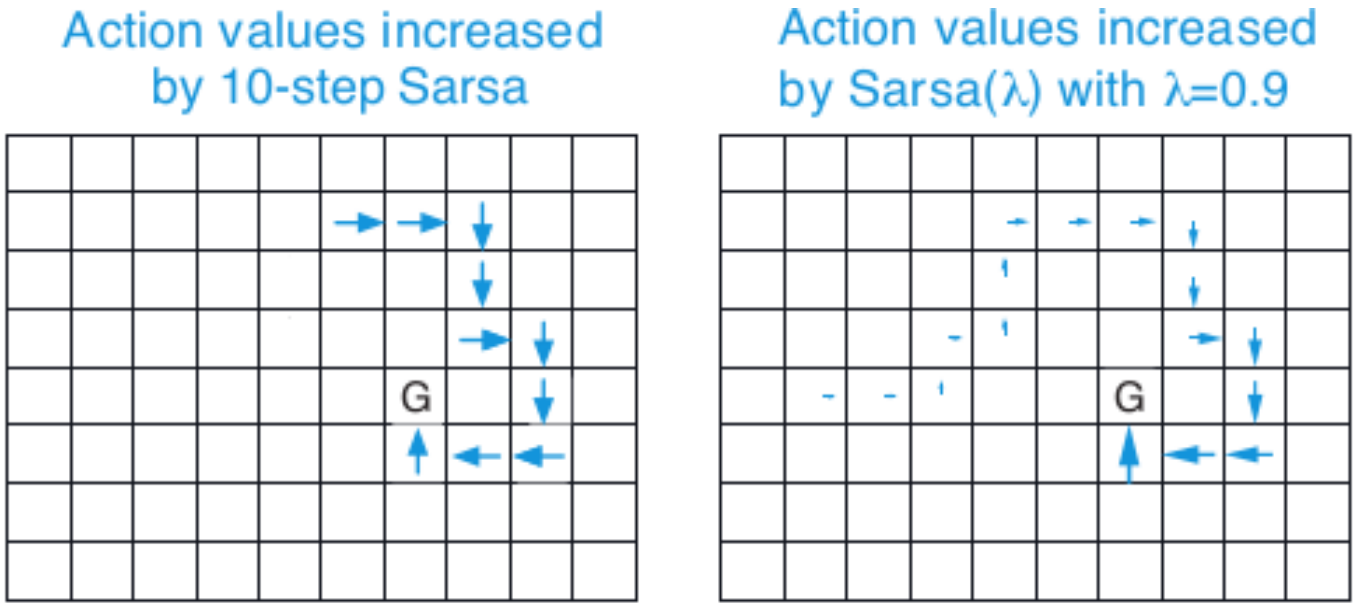
$$\delta_t = R_{t+1} + \gamma \hat{q}(S_{t+1}, A_{t+1}, \bar{w}_t) - \hat{q}(S_t, A_t, \bar{w}_t)$$
$$\bar{w}_{t+1} = \bar{w}_t + \alpha \delta_t \bar{z}_t$$

Algoritmus Sarsa(λ) odhadu $\hat{q}_\pi$ alebo $\hat{q}_*$

Input: a feature function $x : S^+ \to R^d$ such that $x(\text{terminal},\cdot) = 0$
Input: a policy π (if estimating $\hat{q}_\pi$)
Algorithm parameters: step size $\alpha > 0$, trace decay rate λ in $[0, 1]$
Initialize: value-function weights $\bar{w}$ in R^d (e.g., $\bar{w} = 0$)

Loop for each episode:
    Initialize S
    Choose $A \sim \pi(.|S)$ or near greedily from S using $\bar{w}$
    $\bar{z} \leftarrow 0$
    Loop for each step of episode:
        Take action A, observe R, S'
        Choose $A' \sim \pi(.|S')$ or near greedily from S' using $\bar{w}$
        $\bar{z} \leftarrow \gamma\lambda\bar{z} + \nabla\hat{q}(S, A, \bar{w})$
        $\delta \leftarrow R + \gamma\hat{q}(S', A', \bar{w}) - \hat{q}(S, A', \bar{w})$
        $\bar{w} \leftarrow \bar{w} + \alpha\delta\bar{z}$
        $S \leftarrow S', A \leftarrow A'$
    until S' is terminal

Sarsa(n) vs Sarsa(λ)



©*Sutton-Barto: Reinforcement Learning, 2nd ed., 2018*

Orezaná λ-odmena

- Nevýhoda λ-odmeny - vedie na offline algoritmus
- Náhrada orezanou verziou

$$G^{\lambda}_{t:h} = (1-\lambda)\sum_{n=1}^{h-t-1} \lambda^{n-1} G_{t:t+n} + \lambda^{h-t-1} G_{t:h}$$

  - potrebuje dáta (odmeny) iba po nejaký horizont h $(h \leq T)$
    - nie je obmedzené iba na epizodickú úlohu
    - možnosť online algoritmu
  - reziduálnu kumulatívnu odmenu (po horizonte) iba odhaduje najdlhšou kumulatívnou odmenou

Online algoritmus na orezanej λ-odmene

- Princíp
  - inkrementálne bude zbierať dáta po jednom kroku
  - po každom kroku sa vráti a urobí opätovné aktualizácie od začiatku epizódy
- Simulácia $(S_0,\ w_0)$
  - $h = 1 : S_0, A_0, R_1, S_1 \rightarrow G^{\lambda}_{0:1}$
    - $w_1^1 = w_0 + \alpha[G^{\lambda}_{0:1} - \hat{v}(S_0, w_0^1)]\nabla\hat{v}(S_0, w_0^1)$
  - $h = 2 : S_0, A_0, R_1, S_1, A_1, R_2, S_2 \rightarrow G^{\lambda}_{0:2}, G^{\lambda}_{1:2}$
    - $w_1^2 = w_0 + \alpha[G^{\lambda}_{0:2} - \hat{v}(S_0, w_0^2)]\nabla\hat{v}(S_0, w_0^2)$
    - $w_2^2 = w_1^2 + \alpha[G^{\lambda}_{1:2} - \hat{v}(S_1, w_1^2)]\nabla\hat{v}(S_1, w_1^2)$
  - $h = 3 : S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, S_3 \rightarrow G^{\lambda}_{0:3}, G^{\lambda}_{1:3}, G^{\lambda}_{2:3}$
    - $w_1^3 = w_0 + \alpha[G^{\lambda}_{0:3} - \hat{v}(S_0, w_0^3)]\nabla\hat{v}(S_0, w_0^3)$
    - $w_2^3 = w_1^3 + \alpha[G^{\lambda}_{1:3} - \hat{v}(S_1, w_1^3)]\nabla\hat{v}(S_1, w_1^3)$
    - $w_3^3 = w_2^3 + \alpha[G^{\lambda}_{2:3} - \hat{v}(S_2, w_2^3)]\nabla\hat{v}(S_2, w_2^3)$

True online TD(λ) algoritmus

- Predchádzajúci algoritmus je veľmi výpočtovo náročný $\rightarrow$ náhrada pomocou ET
  - iba pre prípad, že aproximátor hodnotovej funkcie je lineárny ($\hat{v}(s, \bar{x}) = \bar{w}^T \bar{x}(s)$)
- Stopa spôsobilosti (ET) je $\bar{z}$ ($\mid \bar{z} \mid = \mid \bar{w} \mid$)

$$\bar{z}_{-1} = 0$$
$$\bar{z}_t = \gamma\lambda\bar{z}_{t-1} + (1 - \alpha\gamma\lambda\bar{z}_{t-1}^T\bar{x}(S_t))\bar{x}(S_t)$$

- Aktualizácia jednotlivých zložiek $\bar{w}$

$$\delta_t = R_{t+1} + \gamma\hat{v}(S_{t+1}, \bar{w}_t) - \hat{v}(S_t, \bar{w}_t)$$
$$\bar{w}_{t+1} = \bar{w}_t + \alpha\delta_t\bar{z}_t + \alpha(\bar{w}_t^T\bar{x}(S_t) - \bar{w}_{t-1}^T\bar{x}(S_t))(\bar{z}_t - \bar{x}(S_t))$$

Algoritmus true Sarsa(λ) odhadu q

True online Sarsa(λ) for estimating $\mathbf{w}^\top \mathbf{x} \approx q_\pi$ or q_*

Input: a feature function $\mathbf{x} : \mathcal{S}^+ \times \mathcal{A} \to \mathbb{R}^d$ such that $\mathbf{x}(terminal, \cdot) = \mathbf{0}$
Input: a policy π (if estimating q_π)
Algorithm parameters: step size $\alpha > 0$, trace decay rate $\lambda \in [0, 1]$
Initialize: $\mathbf{w} \in \mathbb{R}^d$ (e.g., $\mathbf{w} = \mathbf{0}$)

Loop for each episode:
  Initialize S
  Choose $A \sim \pi(\cdot|S)$ or near greedily from S using $\mathbf{w}$
  $\mathbf{x} \leftarrow \mathbf{x}(S, A)$
  $\mathbf{z} \leftarrow \mathbf{0}$
  $Q_{old} \leftarrow 0$
  Loop for each step of episode:
  | Take action A, observe R, S'
  | Choose $A' \sim \pi(\cdot|S')$ or near greedily from S' using $\mathbf{w}$
  | $\mathbf{x}' \leftarrow \mathbf{x}(S', A')$
  | $Q \leftarrow \mathbf{w}^\top \mathbf{x}$
  | $Q' \leftarrow \mathbf{w}^\top \mathbf{x}'$
  | $\delta \leftarrow R + \gamma Q' - Q$
  | $\mathbf{z} \leftarrow \gamma\lambda \mathbf{z} + \left(1 - \alpha\gamma\lambda \mathbf{z}^\top \mathbf{x}\right) \mathbf{x}$
  | $\mathbf{w} \leftarrow \mathbf{w} + \alpha(\delta + Q - Q_{old})\mathbf{z} - \alpha(Q - Q_{old})\mathbf{x}$
  | $Q_{old} \leftarrow Q'$
  | $\mathbf{x} \leftarrow \mathbf{x}'$
  | $A \leftarrow A'$
  until S' is terminal

©Sutton-Barto: Reinforcement Learning, 2nd ed., 2018