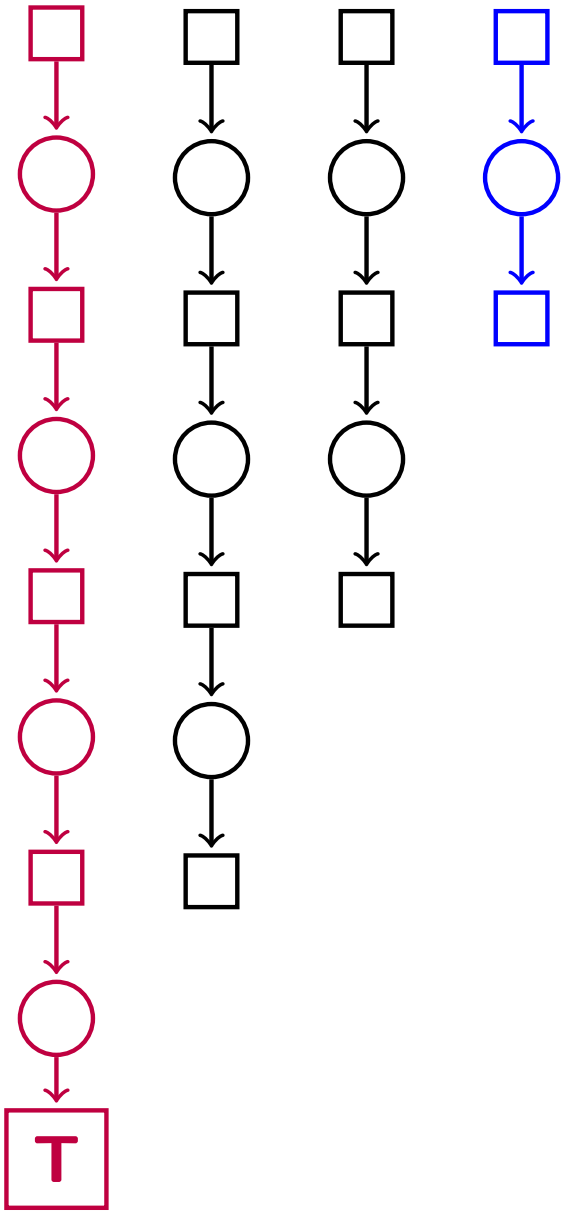
Medzi TD a MC
(Strojové učenie II)

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

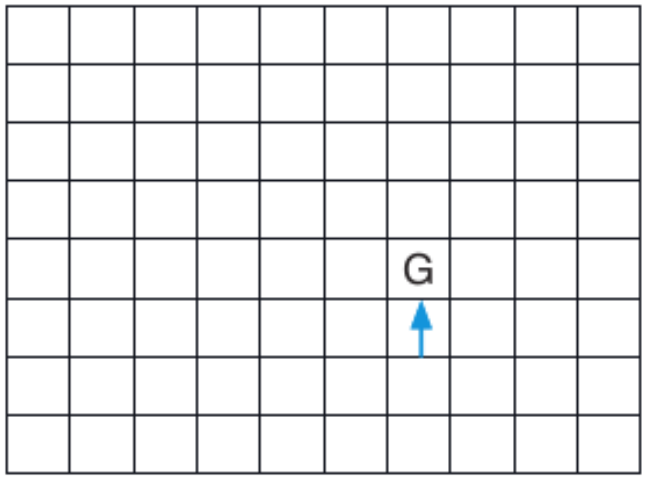
n-step Sarsa for estimating $Q \approx q_*$ or q_π

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 $n+1$

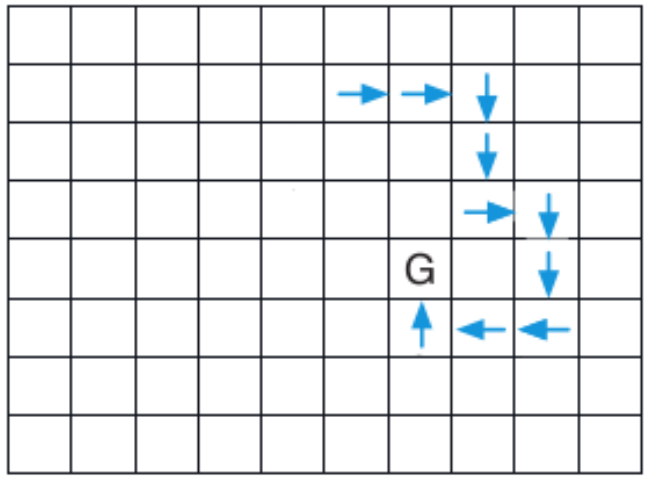
Loop for each episode:
  Initialize and store $S_0 \neq$ terminal
  Select and store an action $A_0 \sim \pi(\cdot|S_0)$
  $T \leftarrow \infty$
  Loop for $t = 0, 1, 2, \ldots$:
    If $t < T$, then:
      Take action A_t
      Observe and store the next reward as R_{t+1} and the next state as S_{t+1}
      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 $\tau = T - 1$

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

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

váhy: $(1-\lambda), (1-\lambda)\lambda, (1-\lambda)\lambda^2, \ldots$

súčet váh: $(1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} = (1-\lambda)\frac{1}{1-\lambda} = 1$

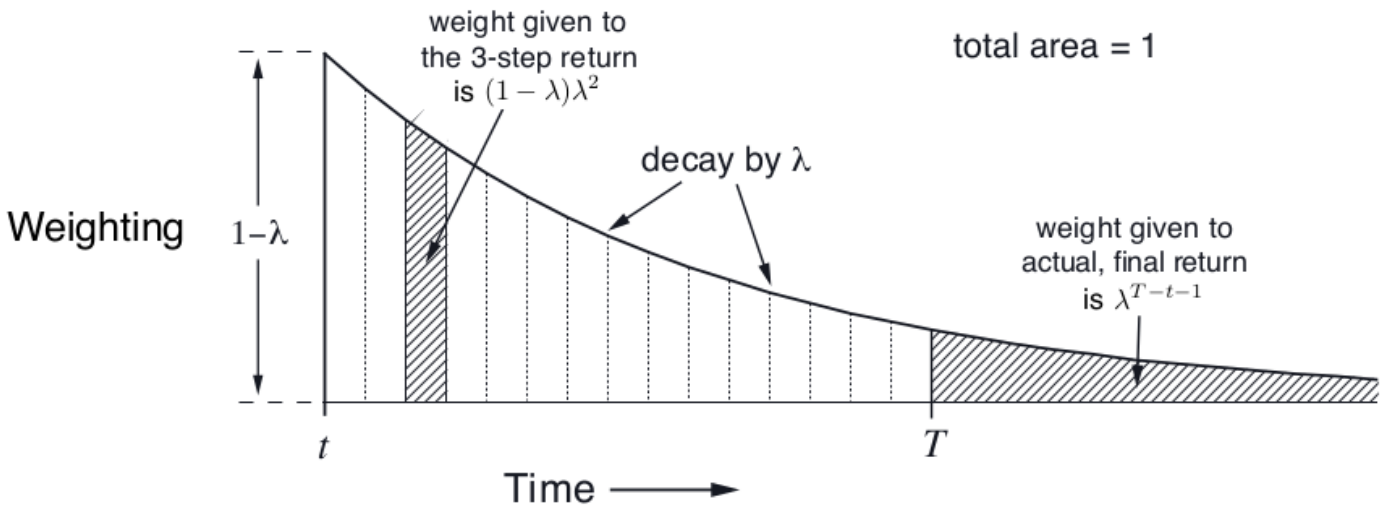
- V prípade epizódy s dĺžkou T ($\max n = T - t$)

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

  - váhy: $(1-\lambda), (1-\lambda)\lambda, \ldots, (1-\lambda)\lambda^{T-t-2}, \lambda^{T-t-1}$
  - $\lambda = 0 \rightarrow G_t^\lambda = G_{t:t+1}$ (TD)
  - $\lambda = 1 \rightarrow G_t^\lambda = G_t$ (MC)

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*

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

Použitie λ-odmeny v algoritme

- Off-line algoritmus
  - aktualizácie robí až po skončení epizódy
  - obmedzenie na epizodickú úlohu
- Aktualizácia pre tabuľkovú reprezentáciu

$$v(S_t) = v(S_t) + \alpha[G_t^\lambda - v(S_t)]$$

- Aktualizácia pre reprezentáciu aproximátorom
  - semi-gradientovým pravidlom

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

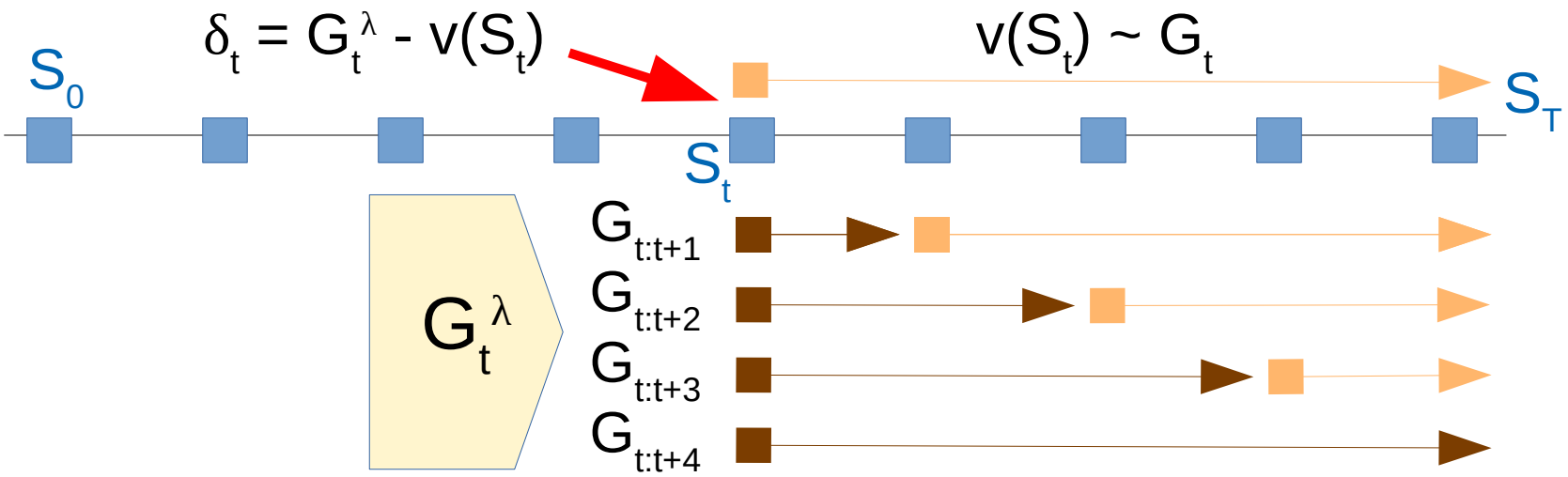
Orezaná λ-odmena

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

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

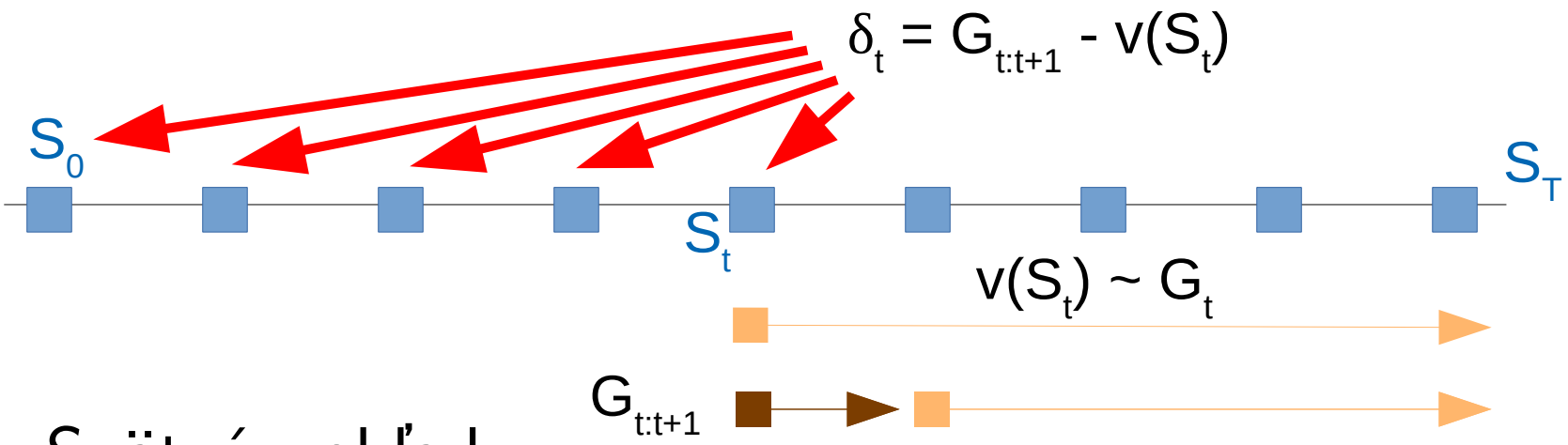
  - potrebuje dáta (odmeny) iba po nejaký horizont h ($h \leq T$)
    - nie je obmedzené iba na epizodickú úlohu
    - možnosť online algoritmu
  - reziduálnu kumulatívnu odmenu (po horizonte) iba odhaduje najdlhšou kumulatívnou odmenou

Dopredný pohľad



- Dopredný pohľad
  - pre každý navštívený stav sa uvažujú budúce odmeny a ich zloženie
  - budúce stavy sú používané opakovane (pre každý z predchádzajúcich stavov)
  - realizovateľné iba ako off-line algoritmus

Spätný pohľad



- Spätný pohľad
  - pre každý navštívený stav sa určí iba $G_{t:t+1}$
  - pre update sa uvažujú aj minulé stavy
  - budúce stavy nie sú používané opakovane
  - realizovateľné ako off-line aj on-line algoritmus
  - založené na eligibility traces
- Dopredný vs spätný pohľad
  - ekvivalentné ak spätný pohľad tiež ako off-line
  - inak podobné ak použité malé α

Eligibility traces - tabuľkový prístup

- Stopa spôsobilosti (ET) je tabuľka $e(s) \mid e(s,a)$
  - ak $v \mid q$ je dlhodobá pamäť (počas celého učenia), tak e je krátkodobá pamäť (iba v rámci epizódy)

$$e_{-1}(s) = 0$$
$$e_t(S_t) = \gamma\lambda e_{t-1}(S_t) + 1$$
$$e_t(s) = \gamma\lambda e_{t-1}(s) \text{ if } s \neq S_t$$

$$e_{-1}(s,a) = 0$$
$$e_t(S_t, A_t) = \gamma\lambda e_{t-1}(S_t, A_t) + 1$$
$$e_t(s,a) = \gamma\lambda e_{t-1}(s,a)$$

- Interpretácia ET
  - pamätá si, cez ktoré stavy (a akcie) sa prechádzalo
  - schopnosť zabúdať minulosť - minulé príspevky pomaly "blednú" faktorom $\gamma\lambda$
- Ovplyvňuje mieru aktualizácie zložiek $v \mid q$

$$\delta_t = R_{t+1} + \gamma v(S_{t+1}) - v(S_t)$$
$$v_{t+1}(s) = v_t(s) + \alpha\delta_t e_t(s) \quad \text{pre všetky stavy } s, \text{ nielen } S_t$$

Koeficient útlmu λ

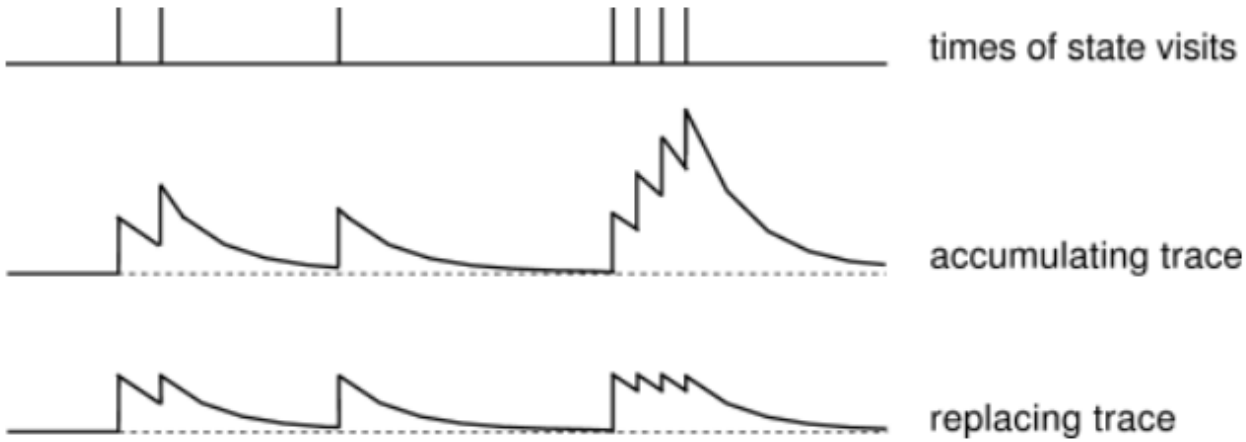
- $\lambda = 0$
  - $e_t(s) = 1$ ak $s = S_t$ inak $e_t(s) = 0$
  - redukcia na jednokrokový TD algoritmus
- $0 < \lambda < 1$
  - čím je hodnota väčšia, tým viac sú predchádzajúce stavy zohľadňované
  - čím je stav hlbšie v minulosti, tým menej je ovplyvňovaný
- $\lambda = 1$
  - minulé stavy sú diskontované iba 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 on-line (nemusí čakať do konca epizódy) - vie reagovať okamžite na novú situáciu

Druhy ET

- Uvedený tvar ET sa označuje ako akumulačný
- Jednoduchším tvarom je nahradzovací
  - iba pre tabuľkovú reprezentáciu alebo binárne príznaky
  - v niektorých úlohách poskytuje rýchlejšiu konvergenciu

$$e_{-1}(s) = 0$$
$$e_t(S_t) = 1$$
$$e_t(s) = \gamma\lambda e_{t-1}(s) \quad \text{if } s \neq S_t$$

$$e_{-1}(s,a) = 0$$
$$e_t(S_t, A_t) = 1$$
$$e_t(s,a) = \gamma\lambda e_{t-1}(s,a)$$



©*Sutton-Barto: Reinforcement Learning, 1998*

Tabular Sarsa(λ) odhadu q_π alebo q_*

Input: a policy π (if estimating q_π) or $\epsilon > 0$ (if π is ϵ-greedy)
Algorithm parameters: step size $\alpha > 0$, trace decay rate λ in $[0, 1]$
Initialize: value-action function $q(s, a)$ arbitrarily for all s, a

Loop for each episode:
- Initialize S
- Choose $A \sim \pi(.|S)$ or near greedily from S (e.g. ϵ-greedy)
- $e(s, a) = 0$ for all s, a
- Loop for each step of episode:
  - Take action A, observe R, S'
  - Choose $A' \sim \pi(.|S')$ or near greedily from S'
  - $e(S_t, A_t) \leftarrow e(S_t, A_t) + 1$
  - $\delta \leftarrow R + \gamma q(S', A') - q(S, A)$
  - for all s, a:
    - $q(s, a) \leftarrow q(s, a) + \alpha\delta e(s, a)$
    - $e(s, a) \leftarrow \gamma\lambda e(s, a)$
  - $S \leftarrow S', A \leftarrow A'$
- until S' is terminal

Eligibility traces - aproximačný prístup

- 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)

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

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

- 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*

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 $\bar{w}$

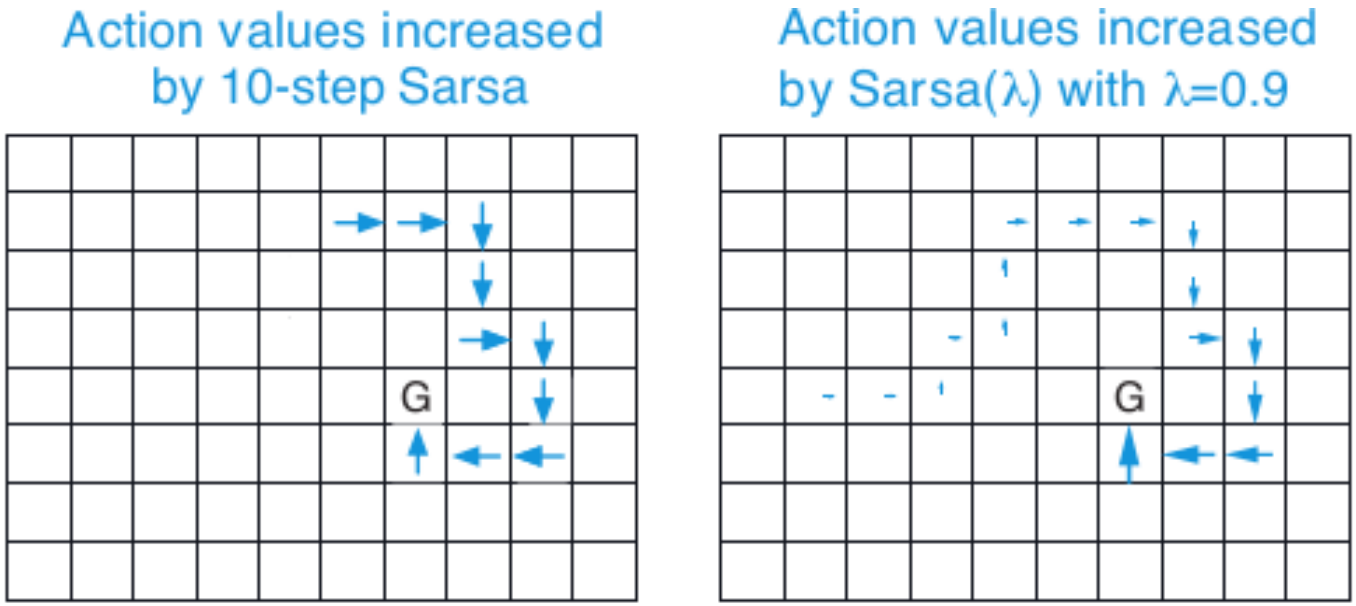
$$\delta_t = R_{t+1} + \gamma \hat{q}(S_{t+1}, A_{t+1}, \bar{w}_t) - \hat{q}(S_t, A_t, \bar{w}_t)$$
$$\bar{w}_{t+1} = \bar{w}_t + \alpha \delta_t \bar{z}_t$$

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*