Lecture 6: Eligibility Traces

Lei Zhang

Outline

- Prediction: TD(λ)
- Control: Sarsa(λ) & Q(λ)
- The Unified View of RL Solutions

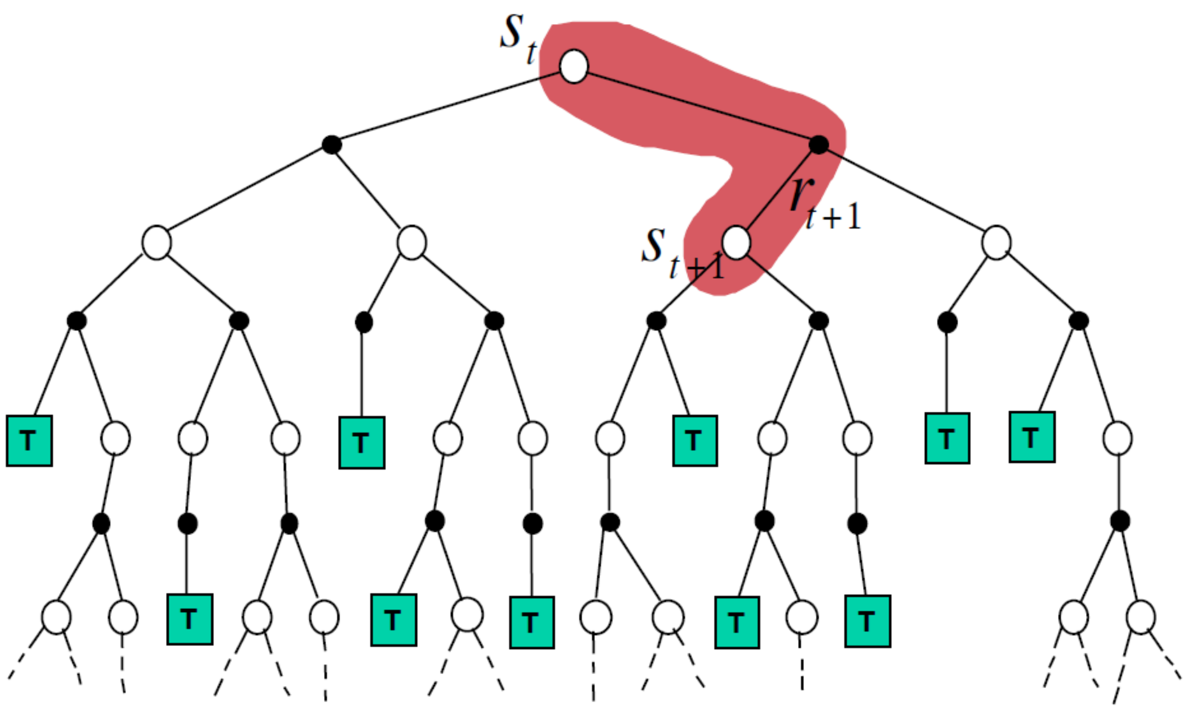
*Materials are modified from David Silver's RL lecture notes

Outline

- Prediction: TD(λ)
- Control: Sarsa(λ) & Q(λ)
- The Unified View of RL Solutions

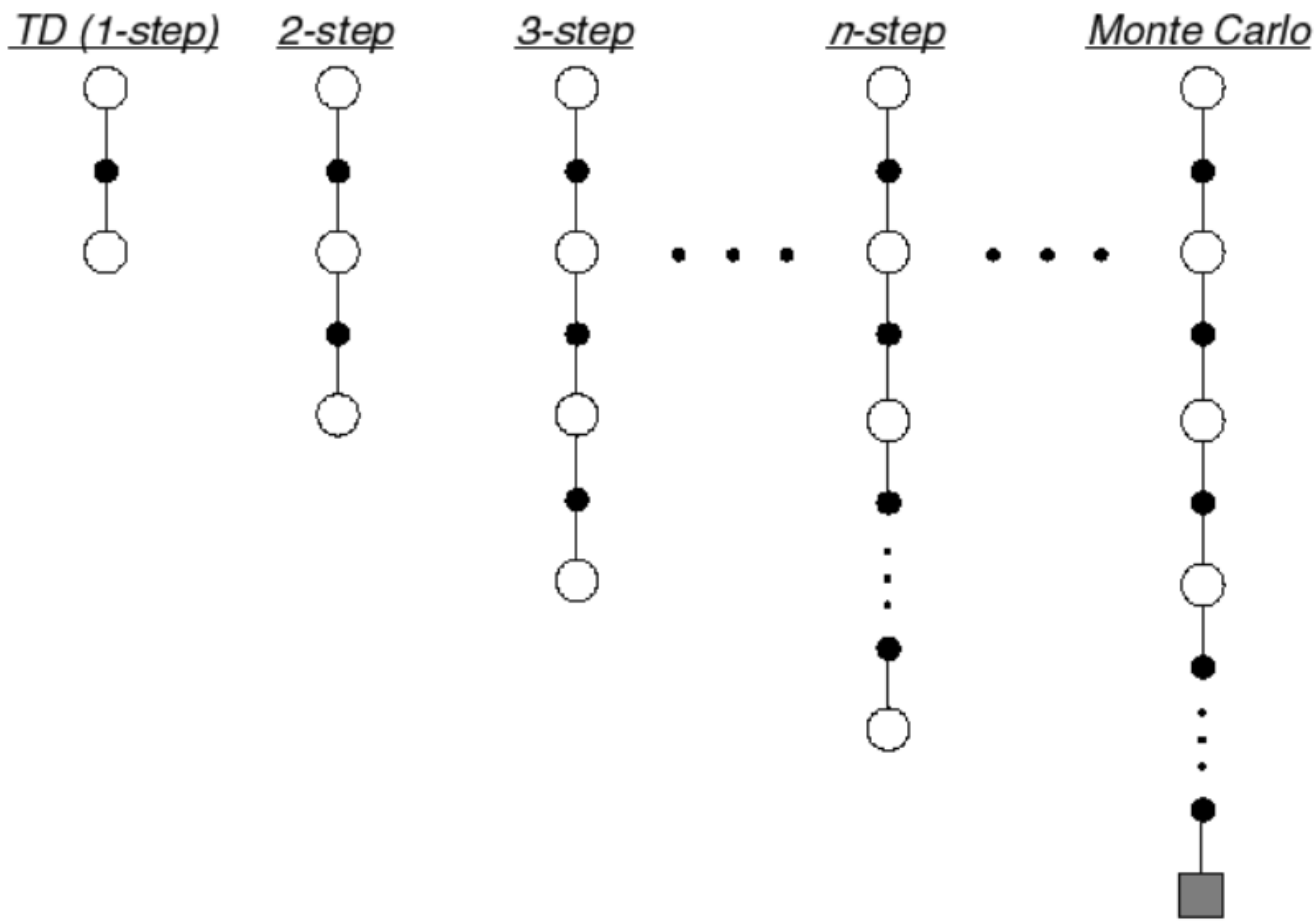
TD(0) Backup (Refresher)

$$V(S_t) \leftarrow V(S_t) + \alpha\left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right)$$



n-step prediction

Let TD target look n steps into the future



n-step return

- Consider the following n-step returns for $n = 1, 2, \infty$:

$$\begin{array}{lll} n = 1 & (TD) & G_t^{(1)} = R_{t+1} + \gamma V(S_{t+1}) \\ n = 2 & & G_t^{(2)} = R_{t+1} + \gamma R_{t+2} + \gamma^2 V(S_{t+2}) \\ \vdots & & \vdots \\ n = \infty & (MC) & G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{T-1} R_T \end{array}$$

- Define the n-step return

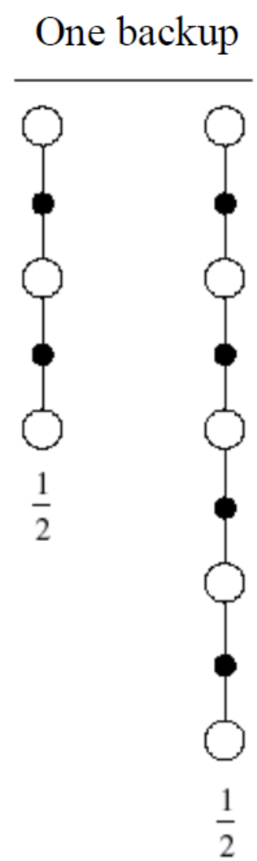
$$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})$$

- n-step temporal-difference learning

$$V(S_t) \leftarrow V(S_t) + \alpha \left(G_t^{(n)} - V(S_t) \right)$$

Averaged n-step return

- We can average n-step returns over different n
  e.g. average the 2-step and 4-step returns
  $$\frac{1}{2}G^{(2)} + \frac{1}{2}G^{(4)}$$
- Combines information from two different time-steps
- Can we efficiently combine information from all time-steps?



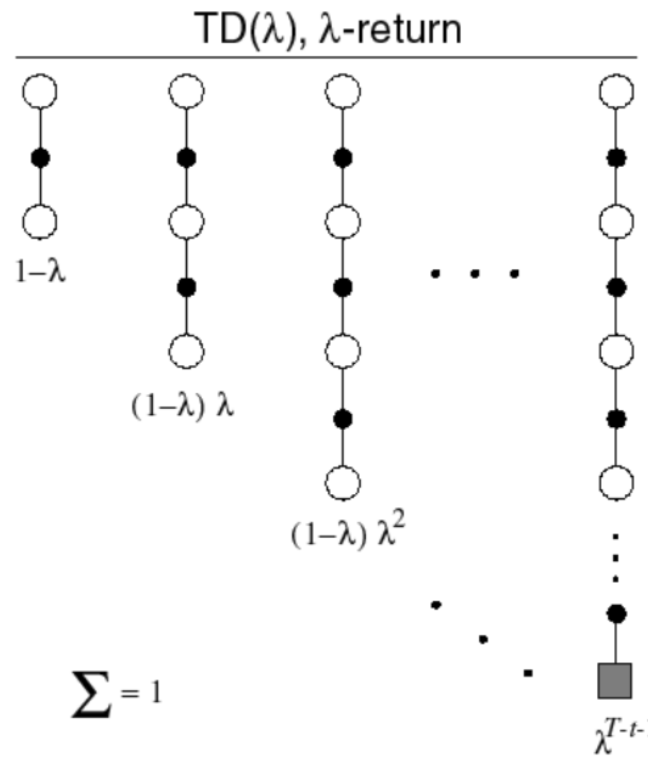
λ -return

- The λ-*return* G_t^λ combines all n-step returns $G_t^{(n)}$
- Using weight $(1-\lambda)\lambda^{n-1}$

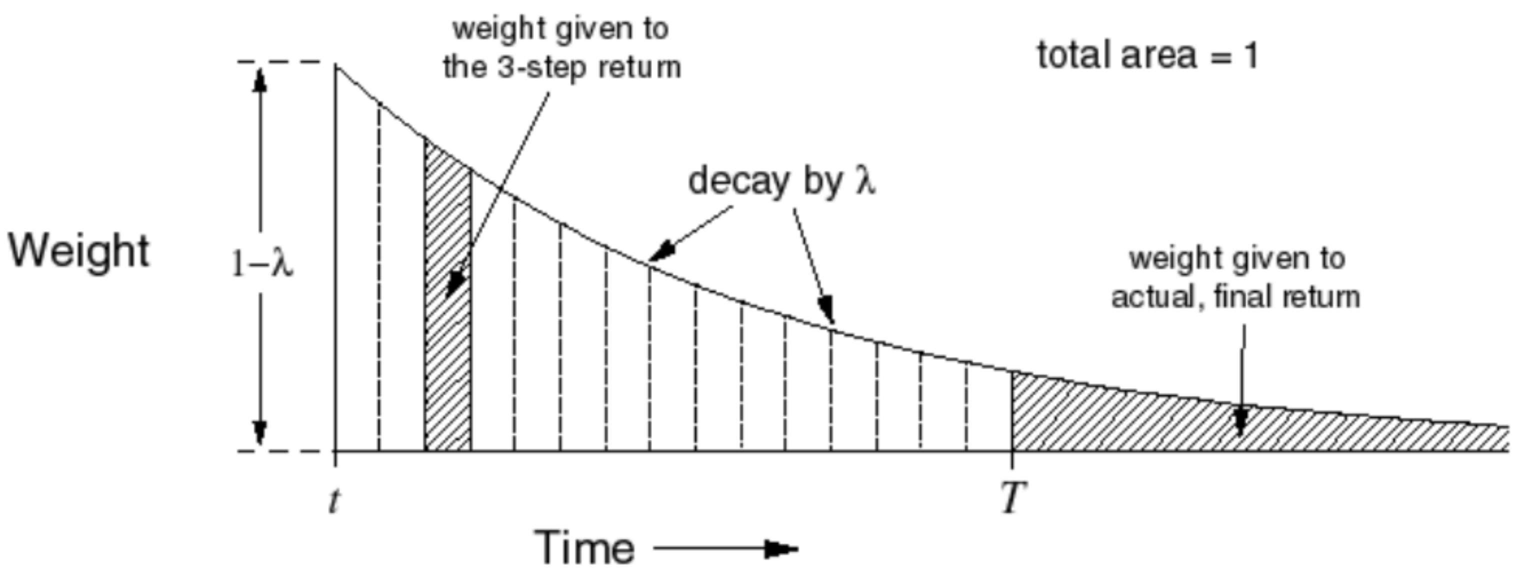
$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_t^{(n)}$$

- Forward-view TD(λ)

$$V(S_t) \leftarrow V(S_t) + \alpha\left(G_t^\lambda - V(S_t)\right)$$

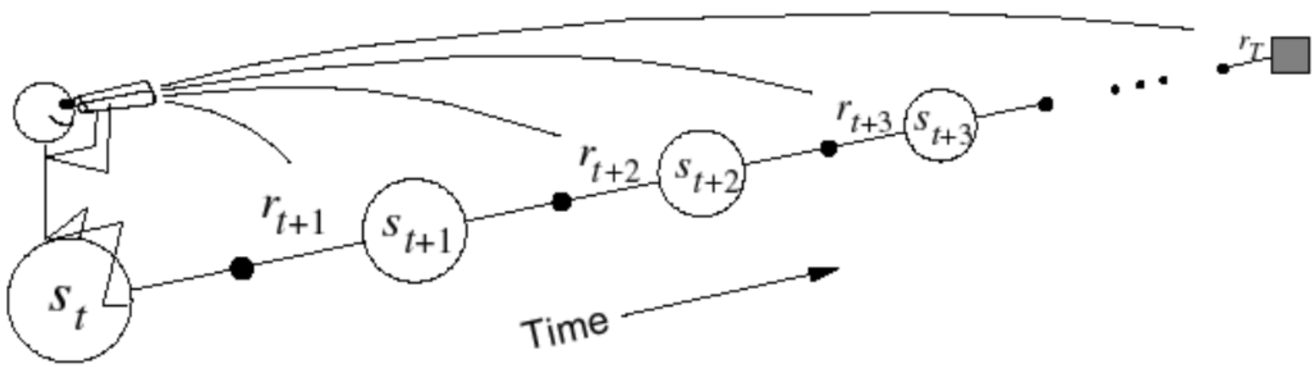


Weighting Function



$$G_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_t^{(n)}$$

Forward View TD(λ)



- Update value function towards the λ-return
- Forward-view looks into the future to compute G_t^λ
- Like MC, can only be computed from complete episodes

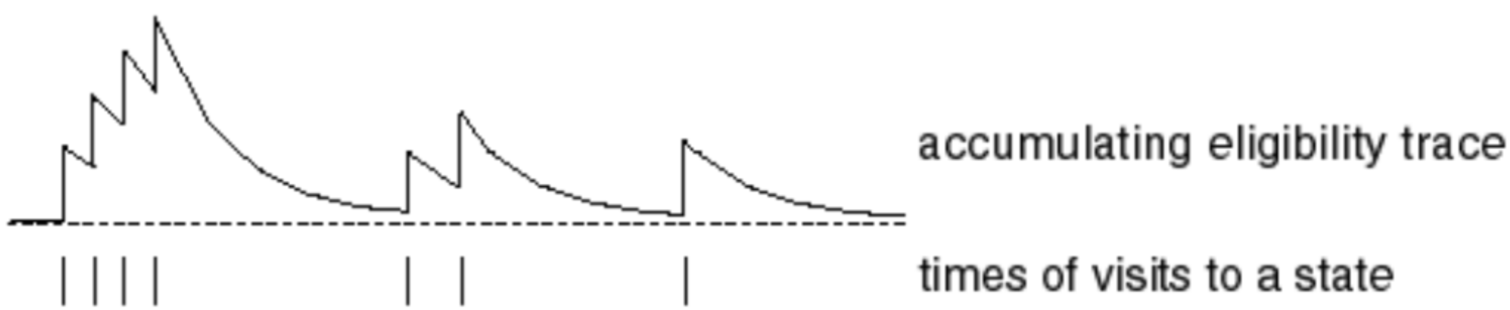
Forward View & Backward View

- Forward view provides theory
- Backward view provides mechanism
- Update online, every step, from incomplete sequences

Eligibility Trace

- Frequency heuristic: assign credit to most frequent states
- Recency heuristic: assign credit to most recent states
- *Eligibility traces* combine both heuristics

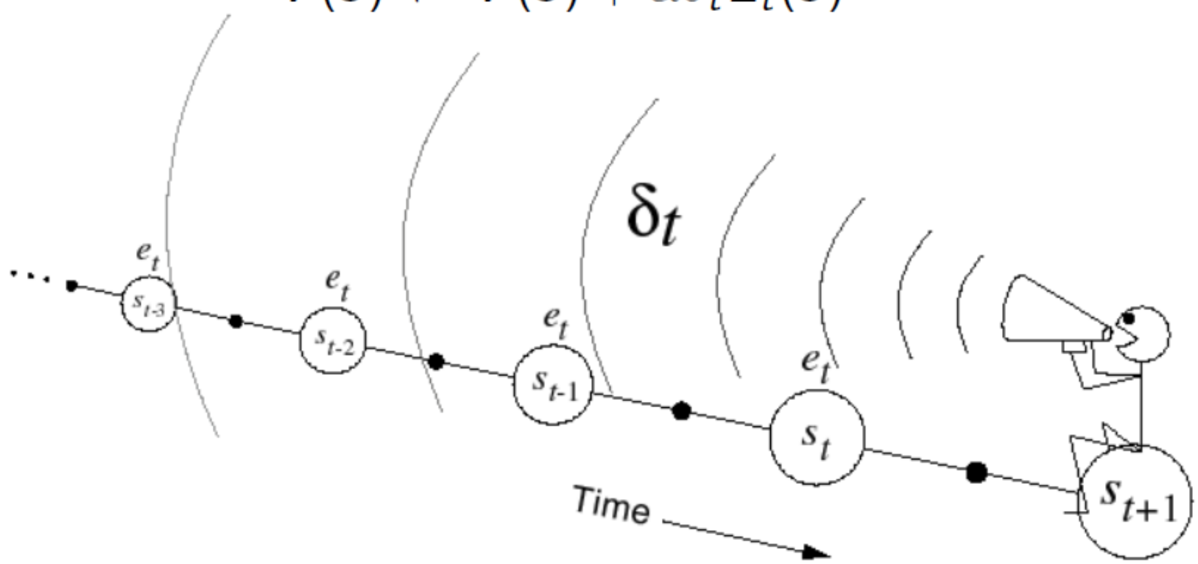
$$E_0(s) = 0$$
$$E_t(s) = \gamma\lambda E_{t-1}(s) + \mathbf{1}(S_t = s)$$



Backward View TD(λ)

- Keep an eligibility trace for every state s
- Update value $V(s)$ for every state s
- In proportion to TD-error δ_t and eligibility trace $E_t(s)$

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$$
$$V(s) \leftarrow V(s) + \alpha \delta_t E_t(s)$$



TD(λ) and TD(0)

- When $\lambda = 0$, only current state is updated

$$E_t(s) = \mathbf{1}(S_t = s)$$
$$V(s) \leftarrow V(s) + \alpha \delta_t E_t(s)$$

- This is exactly equivalent to TD(0) update

$$V(S_t) \leftarrow V(S_t) + \alpha \delta_t$$

Offline Equivalence of Forward and Backward Views

- Updates are accumulated within episode
- but applied in batch at the end of episode

The sum of offline updates is identical for forward-view and backward-view TD(λ)

$$\sum_{t=1}^{T} \alpha \delta_t E_t(s) = \sum_{t=1}^{T} \alpha \left(G_t^{\lambda} - V(S_t) \right) \mathbf{1}(S_t = s)$$

On-line TD(λ)

Initialize $V(s)$ arbitrarily
Repeat (for each episode):
  Initialize $Z(s) = 0$, for all $s \in \mathcal{S}$
  Initialize S
  Repeat (for each step of episode):
    $A \leftarrow$ action given by π for S
    Take action A, observe reward, R, and next state, S'
    $\delta \leftarrow R + \gamma V(S') - V(S)$
    $Z(S) \leftarrow Z(S) + 1$
    For all $s \in \mathcal{S}$:
      $V(s) \leftarrow V(s) + \alpha \delta Z(s)$
      $Z(s) \leftarrow \gamma \lambda Z(s)$
    $S \leftarrow S'$
  until S is terminal

Notation: $Z(s) = E(s)$

- No equivalence b/w Backward and Forward.
- Step size is sufficiently small -> almost “equivalence”
- Online TD updates are generally better than off-line TD.

Online Equivalence of Forward and Backward Views

- *True on-line TD(λ)* :
  - Van Seijen & Sutton, "True Online TD(λ)" ICML 2014
  - Van Seijen, etc "True Online TD Learning", JMLR, 2016

- Forward View: truncated λ-return

$$G_t^{\lambda} = (1-\lambda)\sum_{n=1}^{t'-1} \lambda^{n-1} G_t^{(n)} + \lambda^{t'-1} G_t^{(t')}$$

- Backward View: a slightly different form of eligibility trace

Summary

Offline updates	$\lambda = 0$	$\lambda \in (0, 1)$	$\lambda = 1$
Backward view	TD(0)	TD(λ)	TD(1)
	$\parallel$	$\parallel$	$\parallel$
Forward view	TD(0)	Forward TD(λ)	MC
Online updates	$\lambda = 0$	$\lambda \in (0, 1)$	$\lambda = 1$
Backward view	TD(0)	TD(λ)	TD(1)
	$\parallel$	$\nparallel$	$\nparallel$
Forward view	TD(0)	Forward TD(λ)	MC
	$\parallel$	$\parallel$	$\parallel$
Exact Online	TD(0)	Exact Online TD(λ)	Exact Online TD(1)

Proof?

$=$ here indicates equivalence in total update at end of episode.

Outline

- Prediction: TD(λ)
- Control: Sarsa(λ) & Q(λ)
- The Unified View of RL Solutions

n-step Sarsa

- Consider the following n-step returns for $n = 1, 2, \infty$:

$$\begin{array}{lll} n = 1 & (Sarsa) & q_t^{(1)} = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) \\ n = 2 & & q_t^{(2)} = R_{t+1} + \gamma R_{t+2} + \gamma^2 Q(S_{t+2}, A_{t+2}) \\ \vdots & & \vdots \\ n = \infty & (MC) & q_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + ... + \gamma^{T-1} R_T \end{array}$$

- Define the n-step Q-return

$$q_t^{(n)} = R_{t+1} + \gamma R_{t+2} + ... + \gamma^{n-1} R_{t+n} + \gamma^n Q(S_{t+n}, A_{t+n})$$

- n-step Sarsa updates $Q(s, a)$ towards the n-step Q-return

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left(q_t^{(n)} - Q(S_t, A_t) \right)$$

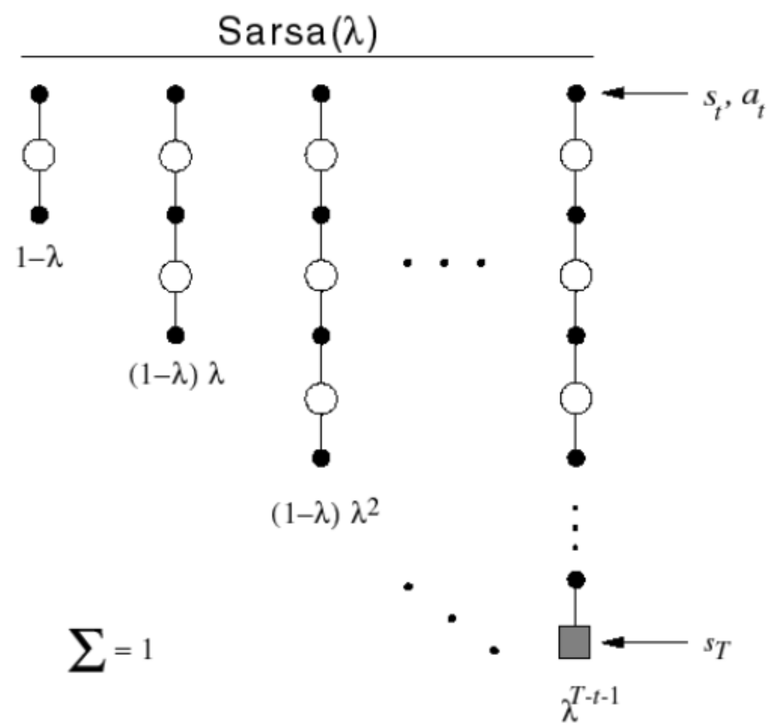
Forward View Sarsa(λ)

- The q^{λ} *return* combines all *n*-step Q-returns $q_t^{(n)}$
- Using weight $(1-\lambda)\lambda^{n-1}$

$$q_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} q_t^{(n)}$$

- Forward-view Sarsa(λ)

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left(q_t^{\lambda} - Q(S_t, A_t)\right)$$



Backward View Sarsa(λ)

- Just like TD(λ), we use eligibility traces in an online algorithm
- But Sarsa(λ) has one eligibility trace for each state-action pair

$$E_0(s, a) = 0$$
$$E_t(s, a) = \gamma\lambda E_{t-1}(s, a) + \mathbf{1}(S_t = s, A_t = a)$$

- $Q(s, a)$ is updated for every state s and action a
- In proportion to TD-error δ_t and eligibility trace $E_t(s, a)$

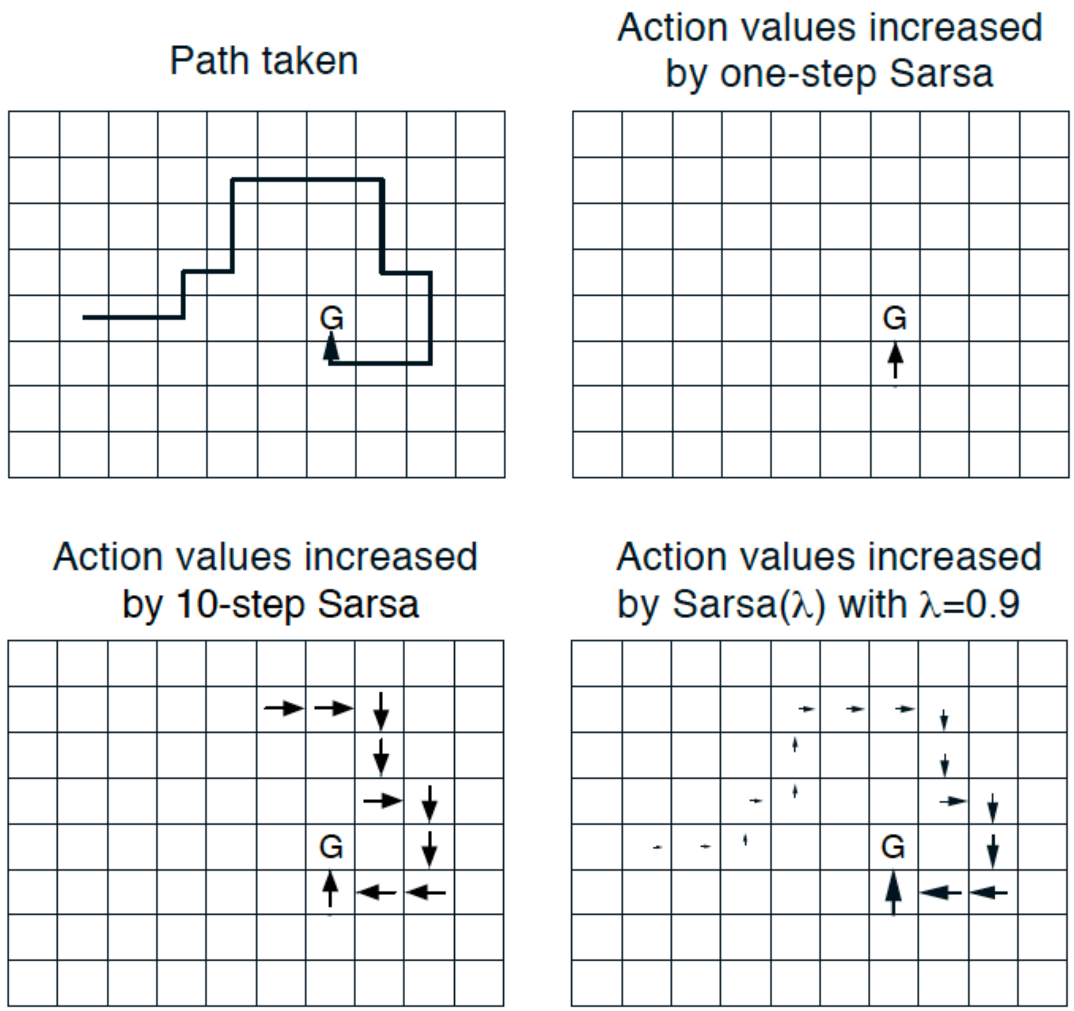
$$\delta_t = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)$$
$$Q(s, a) \leftarrow Q(s, a) + \alpha\delta_t E_t(s, a)$$

Sarsa (λ) Algorithm

Initialize $Q(s,a)$ arbitrarily, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
Repeat (for each episode):
    $Z(s,a) = 0$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
    Initialize S, A
    Repeat (for each step of episode):
        Take action A, observe R, S'
        Choose A' from S' using policy derived from Q (e.g., ε-greedy)
        $\delta \leftarrow R + \gamma Q(S',A') - Q(S,A)$
        $Z(S,A) \leftarrow Z(S,A) + 1$
        For all $s \in \mathcal{S}, a \in \mathcal{A}(s)$:
            $Q(s,a) \leftarrow Q(s,a) + \alpha \delta Z(s,a)$
            $Z(s,a) \leftarrow \gamma \lambda Z(s,a)$
        $S \leftarrow S'$; $A \leftarrow A'$
    until S is terminal

Notation: $Z(s,a) = E(s,a)$

Sarsa (λ) Gridworld Example

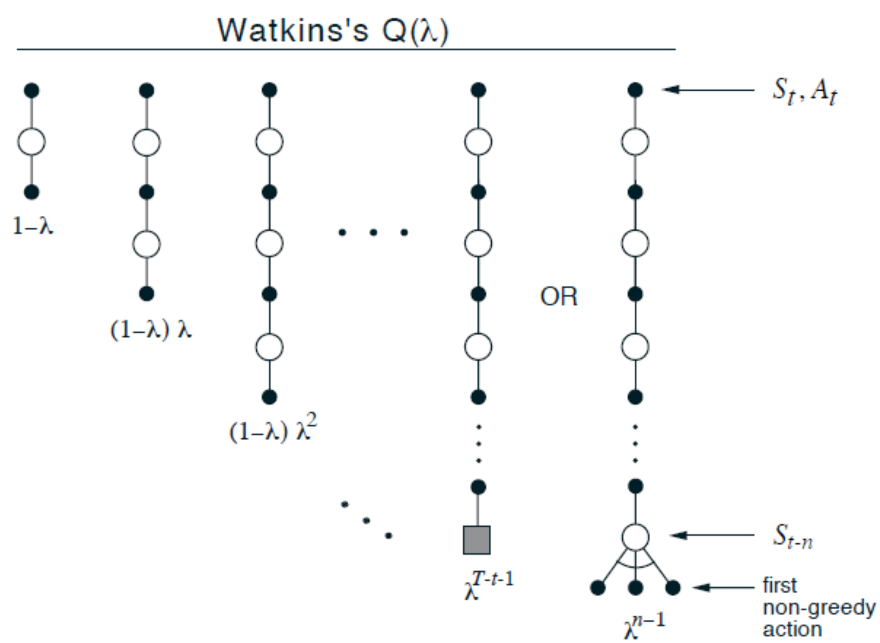


Q (λ) Algorithm – Forward View

- We show Watkins's Q(λ). See literature for others Q(λ) algorithms
- n-step Q return: lookahead stops at the first non-greedy action (i.e. exploratory action). If action at *t+n* is the first non-greedy action, then the longest backup is toward

$$R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n \max_a Q_t(S_{t+n}, a)$$

- Backup diagram



Q (λ) Algorithm – Backward View

- One eligibility trace for each state-action pair

$$E_t(s,a) = \mathbf{1}(S_t = s, A_t = a) + \gamma\lambda E_{t-1}(s,a) \times \mathbf{1}(Q_{t-1}(S_t, A_t) = \max_a Q_{t-1}(S_t, a))$$

- Q update

$$\delta_t = R_{t+1} + \gamma \max_{a'} Q_t(S_{t+1}, a') - Q_t(S_t, A_t)$$

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha\delta_t E_t(s,a)$$

Q (λ) Algorithm

Initialize $Q(s,a)$ arbitrarily, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
Repeat (for each episode):
  $Z(s,a) = 0$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
  Initialize S, A
  Repeat (for each step of episode):
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $A^* \leftarrow \arg\max_a Q(S', a)$ (if A' ties for the max, then $A^* \leftarrow A'$)
    $\delta \leftarrow R + \gamma Q(S', A^*) - Q(S, A)$
    $Z(S,A) \leftarrow Z(S,A) + 1$
    For all $s \in \mathcal{S}, a \in \mathcal{A}(s)$:
      $Q(s,a) \leftarrow Q(s,a) + \alpha\delta Z(s,a)$
      If $A' = A^*$, then $Z(s,a) \leftarrow \gamma\lambda Z(s,a)$
              else $Z(s,a) \leftarrow 0$
    $S \leftarrow S'$; $A \leftarrow A'$
  until S is terminal

Notation: $Z(s,a) = E(s,a)$

Extensions

- Disadvantage of Watkins's Q(λ) ?
- Other Q(λ) algorithms
  - Peng's Q(λ)
  - Naïve Q(λ)

Outline

- Prediction: TD(λ)
- Control: Sarsa(λ) & Q(λ)
- The Unified View of RL Solutions

The Unified View of RL Solutions