Chapter 12

Eligibility Traces

2018/11/27

Dev3B 강준하

Index

Introduction

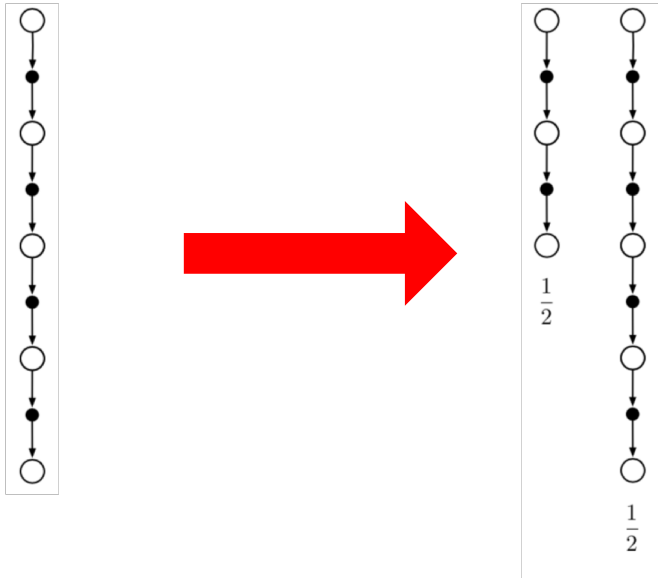
- MC + TD = TD(λ)
- Unify and generalize two methods
- Using a short-term memory vector, the *eligibility trace*
  - Weight is a long-term memory vector
- Pros : Computational advantages, Backward views

12.1 The λ-return

- Recall : n-step return for value function approximation

$$G_{t:t+n} \doteq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n \hat{v}(S_{t+n}, \mathbf{w}_{t+n-1}), \quad 0 \leq t \leq T-n, \quad (12.1)$$

- $\hat{v}(s, \mathbf{w})$: the approximate value of state s given weight vector $\mathbf{w}$
- 4-step return만 보는 게 아니라, 2-step return과 4-step return의 평균을 본다면?

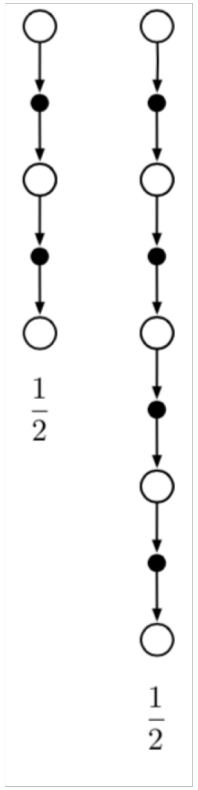


12.1 The λ-return

- $G_{t:t+4} \rightarrow \frac{1}{2} G_{t:t+2} + \frac{1}{2} G_{t:t+4}$
- $G_{t:t+n}$에 대한 일반화를 시도한다면...?

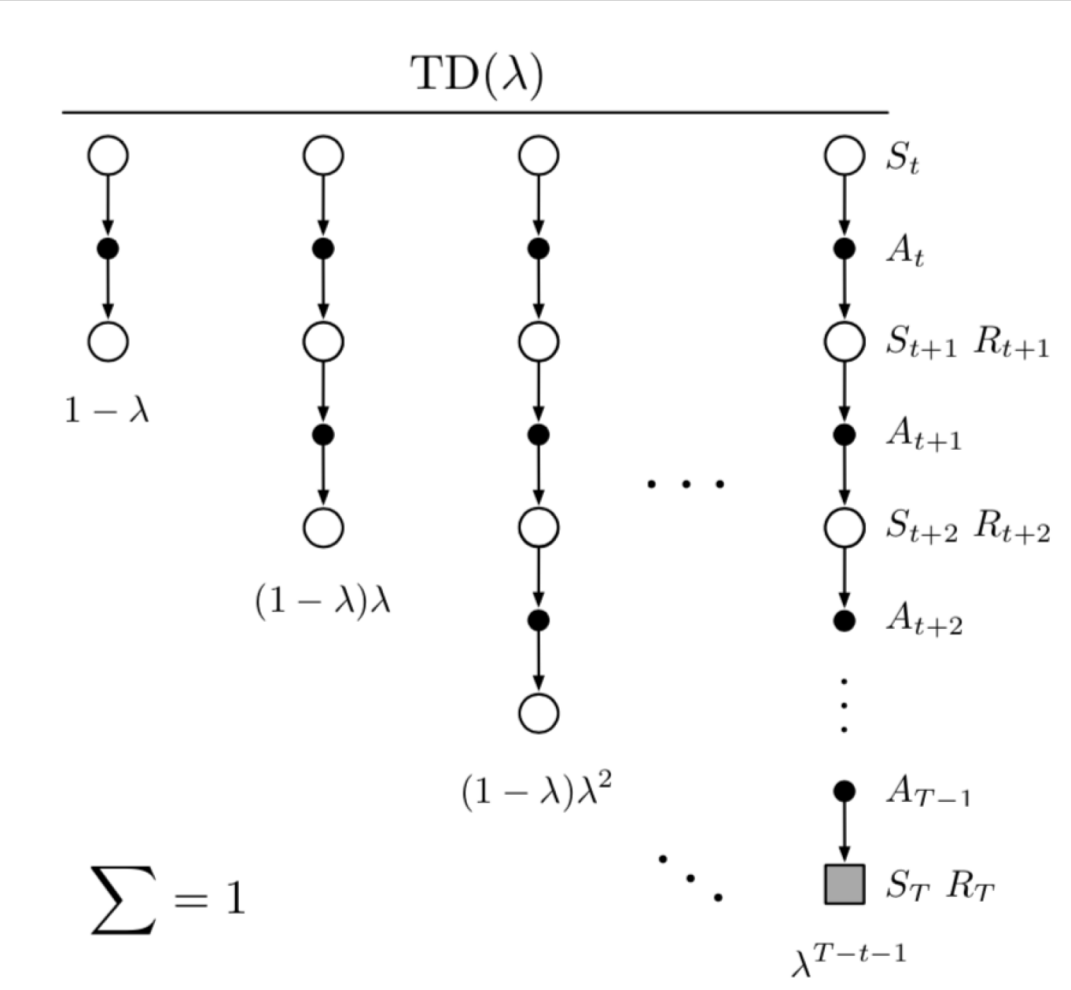
$$G_t^\lambda \doteq (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_{t:t+n}. \qquad (12.2)$$

- Definition of λ-return



12.1 The λ-return

- Backup diagram of λ-return



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

12.1 The λ-return

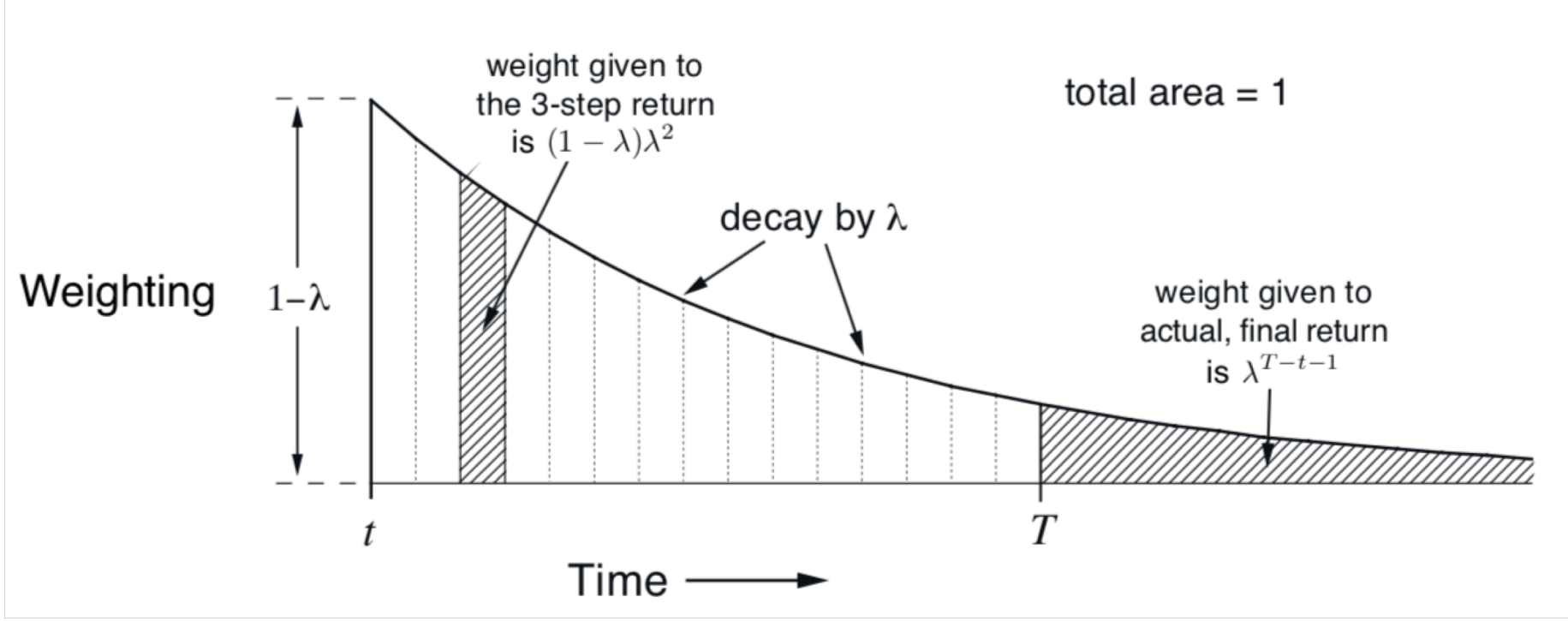
- Weighted average

$$G_t^\lambda \doteq (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_{t:t+n}. \qquad (12.2)$$

$$(1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1} = (1-\lambda)\times\frac{\lambda^0}{1-\lambda} = 1$$

12.1 The λ-return

- Weighting given in the λ-return to each of the n-step returns



$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1}\lambda^{n-1}G_{t:t+n} + \lambda^{T-t-1}G_t, \qquad (12.3)$$

12.1 The λ-return – Exercise 12.1

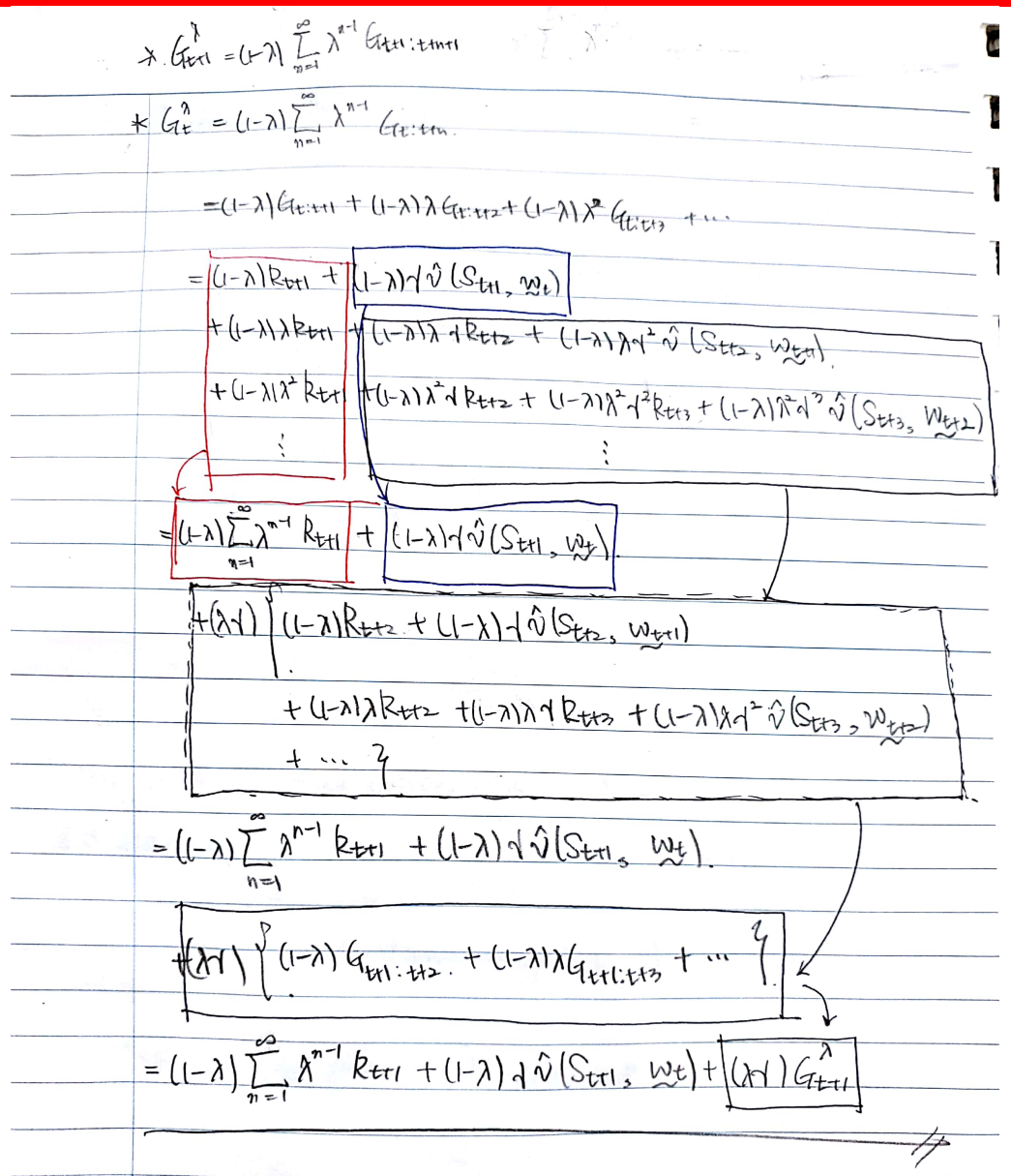
Exercise 12.1 Just as the return can be written recursively in terms of the first reward and itself one-step later (3.9), so can the λ-return. Derive the analogous recursive relationship from (12.2) and (12.1). □

$$\begin{aligned} G_t &\doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots \\ &= R_{t+1} + \gamma\big(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \cdots\big) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned} \tag{3.9}$$

$$G_{t:t+n} \doteq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n \hat{v}(S_{t+n}, \mathbf{w}_{t+n-1}), \quad 0 \le t \le T - n, \tag{12.1}$$

$$G_t^\lambda \doteq (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_{t:t+n}. \tag{12.2}$$

12.1 The λ-return – Exercise 12.1



$$G_t^\lambda = R_{t+1} + \gamma\{(1-\lambda)\hat{v}(S_{t+1}, \mathbf{w}_t) + \lambda G_{t+1}^\lambda\}$$

12.1 The λ-return – Exercise 12.2

Exercise 12.2 The parameter λ characterizes how fast the exponential weighting in Figure 12.2 falls off, and thus how far into the future the λ-return algorithm looks in determining its update. But a rate factor such as λ is sometimes an awkward way of characterizing the speed of the decay. For some purposes it is better to specify a time constant, or half-life. What is the equation relating λ and the half-life, τ_λ, the time by which the weighting sequence will have fallen to half of its initial value? □

- 절반의 영향력을 끼치려면 어디까지 가야 하는가?
- $\lambda^n = \frac{1}{2}$ 가 되는 스텝 수는?
- $\lambda^{\tau_\lambda} = \frac{1}{2}$
- $\tau_\lambda = -\frac{\ln 2}{\ln \lambda}$

12.1 The λ-return

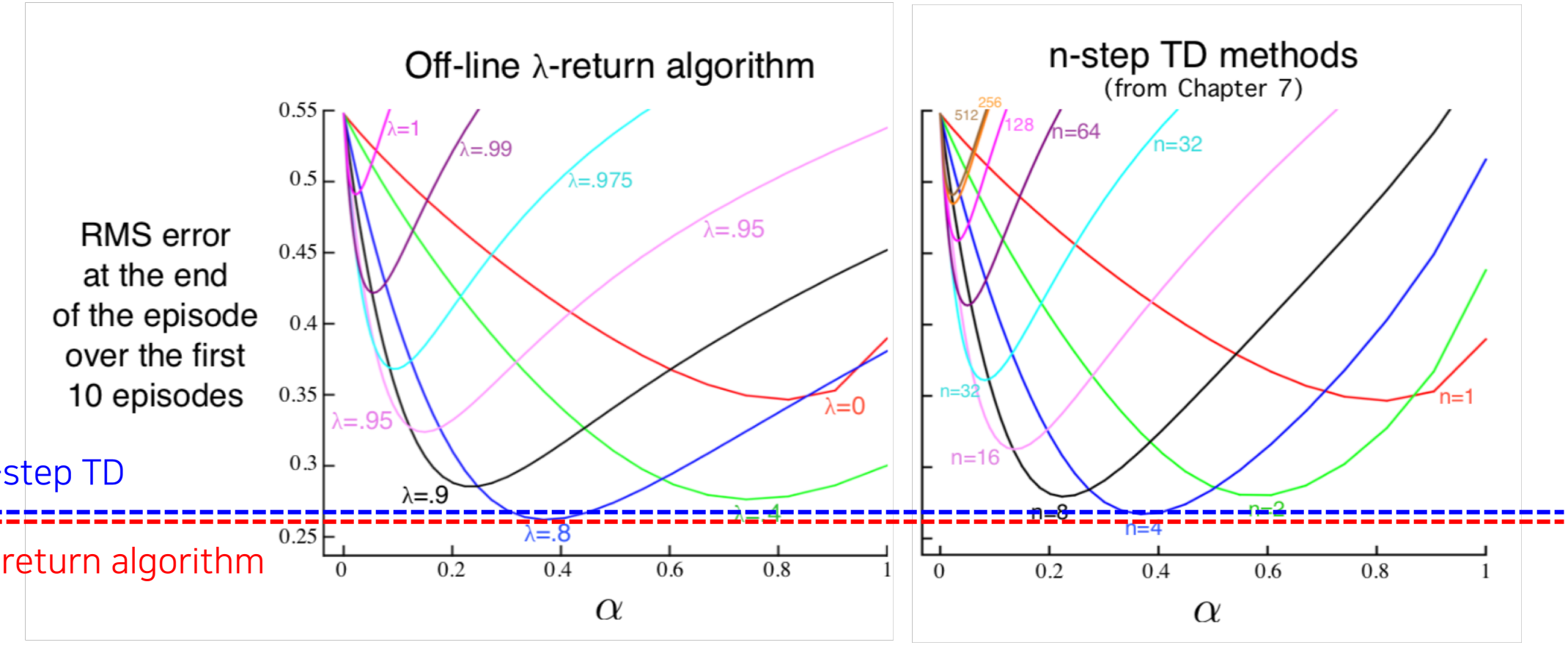
- Offline λ-return algorithm update rule

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\Big[G_t^\lambda - \hat{v}(S_t,\mathbf{w}_t)\Big]\nabla\hat{v}(S_t,\mathbf{w}_t), \quad t = 0,\ldots,T-1. \tag{12.4}$$

- Episode 동안에는 update가 이루어지지 않음
- MC + 1-step TD

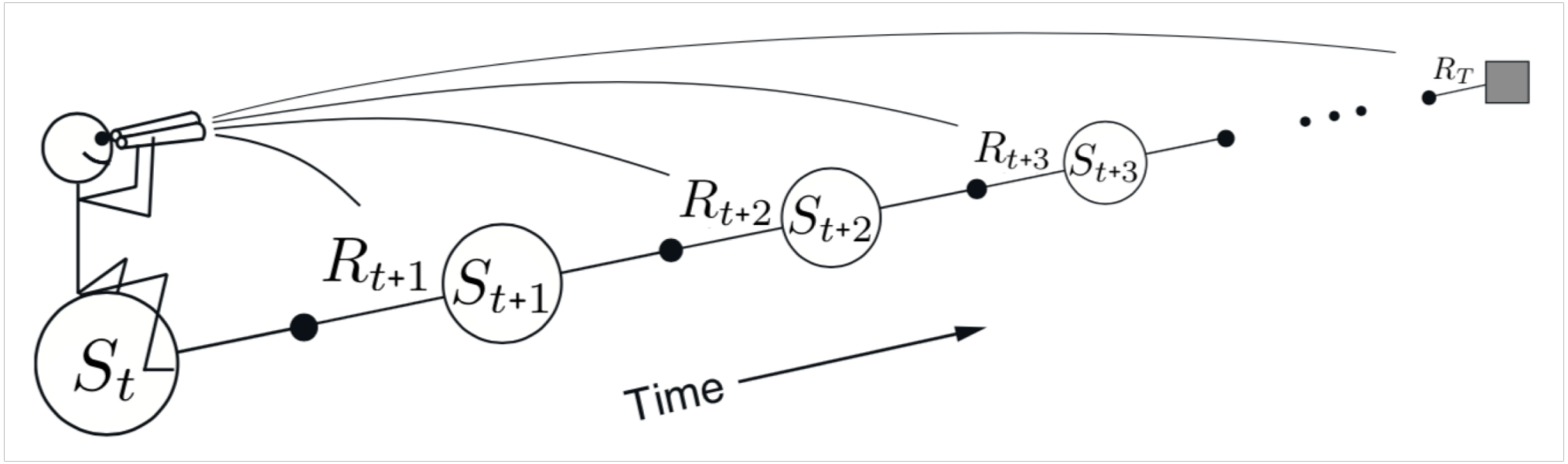
12.1 The λ-return – Figure 12.3

- n-step TD와의 비교 (19 state random walk)



12.1 The λ-return

- The forward view
- 다음 reward와 state를 보고 현재 state의 value를 결정한다.



12.2 TD(λ)

- Recall : offline λ-return algorithm의 update rule

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\Big[G_t^\lambda - \hat{v}(S_t,\mathbf{w}_t)\Big]\nabla\hat{v}(S_t,\mathbf{w}_t), \quad t = 0,\ldots,T-1. \tag{12.4}$$

- G_t^λ의 계산은 episode가 끝나야 가능
- Eligibility trace 이용하여 근사 → TD(λ)
  - Episode가 끝나지 않아도 G_t^λ 근사 가능
  - Computation이 episode가 끝난 시점에 몰리지 않고 균등하게 분배됨

12.2 TD(λ)

- Eligibility trace : vector $\mathbf{z}_t \in \mathbb{R}^d$ (weight vector인 $\mathbf{w}_t$도 같은 차원)

$$\mathbf{z}_{-1} \doteq \mathbf{0},$$
$$\mathbf{z}_t \doteq \gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{v}(S_t,\mathbf{w}_t), \quad 0 \leq t \leq T, \tag{12.5}$$

- true gradient value가 $\gamma\lambda$만큼 vanishing되며 누적 update됨

12.2 TD(λ)

- One-step TD error

$$\delta_t \doteq R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t). \tag{12.6}$$

- update rule

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \delta_t \mathbf{z}_t. \tag{12.7}$$

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

12.2 TD(λ)

- Recall recursive form & offline λ-return algorithm update rule

$$G_t^\lambda = R_{t+1} + \gamma\{(1-\lambda)\hat{v}(S_{t+1}, \mathbf{w}_t) + \lambda G_{t+1}^\lambda\}$$

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\left[G_t^\lambda - \hat{v}(S_t,\mathbf{w}_t)\right]\nabla\hat{v}(S_t,\mathbf{w}_t), \quad t = 0,\ldots,T-1. \tag{12.4}$$

$$\begin{aligned}
\mathbf{w}_{t+1} &\doteq \mathbf{w}_t + \alpha\left[R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}_t) - \gamma\lambda\hat{v}(S_{t+1},\mathbf{w}_t) + \gamma\lambda G_{t+1}^\lambda - \hat{v}(S_{t+1},\mathbf{w}_t)\right]\nabla\hat{v}(S_t,\mathbf{w}_t) \\
&= \mathbf{w}_t + \alpha[R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}_t) - \hat{v}(S_{t+1},\mathbf{w}_t)]\nabla\hat{v}(S_t,\mathbf{w}_t) \\
&\quad + \alpha\gamma\lambda\left[G_{t+1}^\lambda - \hat{v}(S_{t+1},\mathbf{w}_t)\right]\nabla\hat{v}(S_t,\mathbf{w}_t)
\end{aligned}$$

12.2 TD(λ)

- TD(λ) update rule

$$\mathbf{z}_{-1} \doteq \mathbf{0},$$
$$\mathbf{z}_t \doteq \gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{v}(S_t,\mathbf{w}_t), \quad 0 \le t \le T, \tag{12.5}$$

$$\delta_t \doteq R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}_t) - \hat{v}(S_t,\mathbf{w}_t). \tag{12.6}$$

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\delta_t\mathbf{z}_t. \tag{12.7}$$

$$\begin{aligned}\mathbf{w}_{t+1} &\doteq \mathbf{w}_t + \alpha[R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}_t) - \hat{v}(S_{t+1},\mathbf{w}_t)][\gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{v}(S_t,\mathbf{w}_t)] \\ &= \mathbf{w}_t + \alpha[R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}_t) - \hat{v}(S_{t+1},\mathbf{w}_t)]\nabla\hat{v}(S_t,\mathbf{w}_t) \\ &\quad + \alpha\gamma\lambda[R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}_t) - \hat{v}(S_{t+1},\mathbf{w}_t)]\mathbf{z}_{t-1}\end{aligned}$$

12.2 TD(λ)

Offline λ-return algorithm

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\left[R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \gamma\lambda\hat{v}(S_{t+1}, \mathbf{w}_t) + \gamma\lambda G^{\lambda}_{t+1} - \hat{v}(S_{t+1}, \mathbf{w}_t)\right]\nabla\hat{v}(S_t, \mathbf{w}_t)$$

$$= \boxed{\mathbf{w}_t + \alpha[R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_{t+1}, \mathbf{w}_t)]\nabla\hat{v}(S_t, \mathbf{w}_t)}$$

$$\boxed{+ \alpha\gamma\lambda\left[G^{\lambda}_{t+1} - \hat{v}(S_{t+1}, \mathbf{w}_t)\right]\nabla\hat{v}(S_t, \mathbf{w}_t)}$$

TD(λ)

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha[R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_{t+1}, \mathbf{w}_t)][\gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{v}(S_t, \mathbf{w}_t)]$$

$$= \boxed{\mathbf{w}_t + \alpha[R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_{t+1}, \mathbf{w}_t)]\nabla\hat{v}(S_t, \mathbf{w}_t)}$$

$$\boxed{+ \alpha\gamma\lambda[R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_{t+1}, \mathbf{w}_t)]\mathbf{z}_{t-1}}$$

12.2 TD(λ)

$$+ \alpha\gamma\lambda\left[G_{t+1}^{\lambda} - \hat{v}(S_{t+1}, \mathbf{w}_t)\right]\nabla\hat{v}(S_t, \mathbf{w}_t)$$

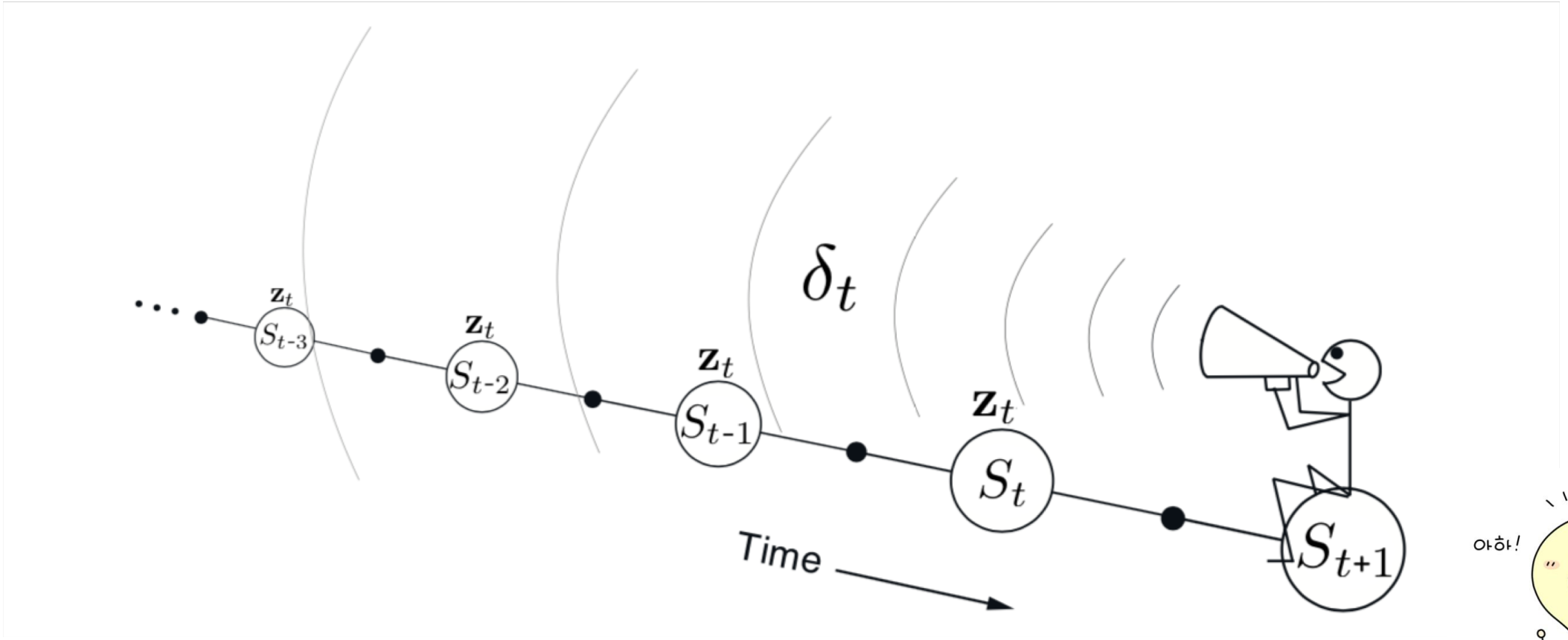
Forward view

$$+ \alpha\gamma\lambda[R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_{t+1}, \mathbf{w}_t)]\mathbf{z}_{t-1}$$

Backward view

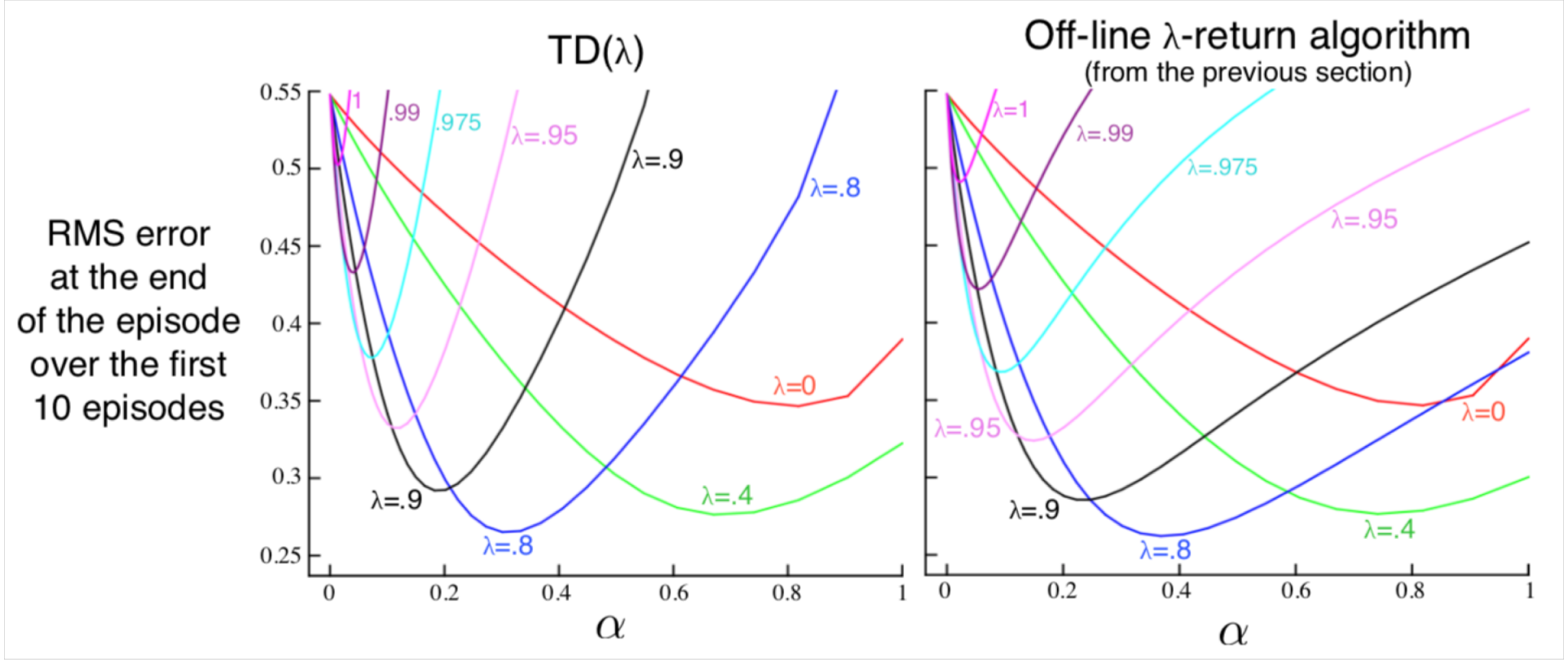
12.2 TD(λ)

- Approximation from the backward view!

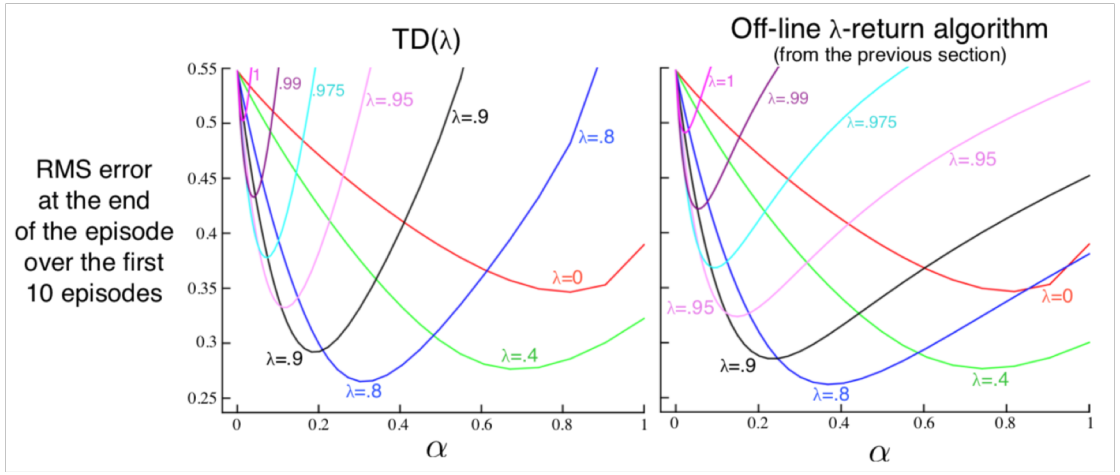


12.2 TD(λ)

- 19 state random walk example



12.2 TD(λ)



- TD(λ)는 높은 α값에서 정상적으로 작동하지 않음
- α값이 곱해지는 update 항의 error가 누적이 빨라지기 때문

12.2 TD(λ)

- At TD(0) (Chapter 9)

$$\overline{\text{VE}}(\mathbf{w}_{\text{TD}}) \leq \frac{1}{1-\gamma} \min_{\mathbf{w}} \overline{\text{VE}}(\mathbf{w}). \tag{9.14}$$

- At TD(λ)

$$\overline{\text{VE}}(\mathbf{w}_{\infty}) \leq \frac{1-\gamma\lambda}{1-\gamma} \min_{\mathbf{w}} \overline{\text{VE}}(\mathbf{w}). \tag{12.8}$$

- TD(λ) also converge

12.2 TD(λ) – Exercise 12.3, 12.4

Exercise 12.3 Some insight into how TD(λ) can closely approximate the offline λ-return algorithm can be gained by seeing that the latter's error term (in brackets in (12.4)) can be written as the sum of TD errors (12.6) for a single fixed $\mathbf{w}$. Show this, following the pattern of (6.6), and using the recursive relationship for the λ-return you obtained in Exercise 12.1. □

Exercise 12.4 Use your result from the preceding exercise to show that, if the weight updates over an episode were computed on each step but not actually used to change the weights ($\mathbf{w}$ remained fixed), then the sum of TD(λ)'s weight updates would be the same as the sum of the offline λ-return algorithm's updates. □

- We already know answer of exercise…

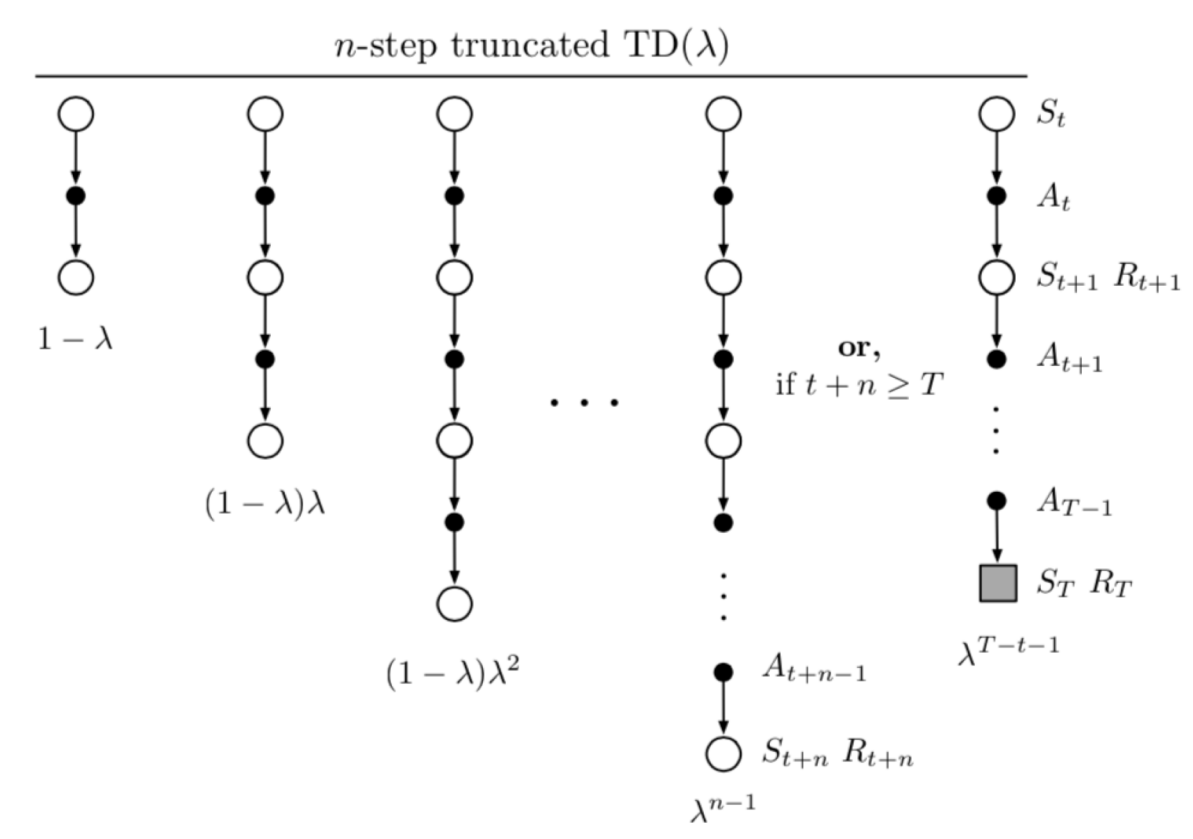
12.3 n-step Truncated λ-return Methods

- Offline λ-return algorithm은 이상적이지만, episode의 끝을 봐야 한다는 점에서 한계가 존재
- Continuous case에서는 update가 불가능
- n-step 이후의 G_t^{λ} 까지만 확인하여 update (이전에 MC에서 horizon과 같이)

12.3 n-step Truncated λ-return Methods

- truncated λ-return (for time t)

$$G_{t:h}^{\lambda} \doteq (1-\lambda)\sum_{n=1}^{h-t-1}\lambda^{n-1}G_{t:t+n} + \lambda^{h-t-1}G_{t:h}, \qquad 0 \le t < h \le T. \qquad (12.9)$$



12.3 n-step Truncated λ-return Methods

- update rule for TTD(λ)

$$\mathbf{w}_{t+n} \doteq \mathbf{w}_{t+n-1} + \alpha \left[G^{\lambda}_{t:t+n} - \hat{v}(S_t, \mathbf{w}_{t+n-1}) \right] \nabla \hat{v}(S_t, \mathbf{w}_{t+n-1}), \qquad 0 \leq t < T.$$

- For efficient implementation,

$$G^{\lambda}_{t:t+k} = \hat{v}(S_t, \mathbf{w}_{t-1}) + \sum_{i=t}^{t+k-1} (\gamma\lambda)^{i-t} \delta'_i, \tag{12.10}$$

$$\delta'_t \doteq R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_{t-1}).$$

12.3 n-step Truncated λ-return Methods

$$G^{\lambda}_{t:t+k} = \hat{v}(S_t, \mathbf{w}_{t-1}) + \sum_{i=t}^{t+k-1} (\gamma\lambda)^{i-t}\delta'_i$$

$$= \hat{v}(S_t, \mathbf{w}_{t-1}) + \delta'_t + (\gamma\lambda)\delta'_{t+1} + (\gamma\lambda)^2\delta'_{t+2} + \cdots + (\gamma\lambda)^{k-1}\delta_{t+k-1}$$

$$= \hat{v}(S_t, \mathbf{w}_{t-1})$$

$$+ R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_{t-1})$$

$$+ (\gamma\lambda)R_{t+2} + \gamma(\gamma\lambda)\hat{v}(S_{t+2}, \mathbf{w}_t) - (\gamma\lambda)\hat{v}(S_{t+1}, \mathbf{w}_{t-1})$$

$$+ (\gamma\lambda)^2R_{t+3} + \gamma(\gamma\lambda)^2\hat{v}(S_{t+3}, \mathbf{w}_t) - (\gamma\lambda)^2\hat{v}(S_{t+2}, \mathbf{w}_{t-1})$$

$$\vdots$$

$$+ (\gamma\lambda)^{k-1}R_{t+k} + \gamma(\gamma\lambda)^{k-1}\hat{v}(S_{t+k}, \mathbf{w}_t) - (\gamma\lambda)^{k-1}\hat{v}(S_{t+k-1}, \mathbf{w}_{t-1})$$

$$= (1-\lambda)[R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t)]$$

$$+ (1-\lambda)\lambda[R_{t+1} + \gamma R_{t+2} + \gamma^2\hat{v}(S_{t+2}, \mathbf{w}_t)]$$

$$\vdots$$

$$+ \lambda^{k-1}[R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^k\hat{v}(S_{t+k}, \mathbf{w}_t)]$$

$$G^{\lambda}_{t:t+k} = \hat{v}(S_t, \mathbf{w}_{t-1}) + \sum_{i=t}^{t+k-1} (\gamma\lambda)^{i-t}\delta'_i,$$

$$\delta'_t \doteq R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_{t-1}).$$

$$1 = (1-\lambda)\sum_{i=1}^{k} \lambda^{i-1} + \lambda^{k-1}$$

12.3 n-step Truncated λ-return Methods

$$
\begin{aligned}
G^{\lambda}_{t:t+k} &= (1-\lambda)[R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}_t)] \\
&\quad + (1-\lambda)\lambda[R_{t+1} + \gamma R_{t+2} + \gamma^2 \hat{v}(S_{t+2}, \mathbf{w}_t)] \\
&\quad \vdots \\
&\quad + \lambda^{k-1}[R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^k \hat{v}(S_{t+k}, \mathbf{w}_t)] \\
&= (1-\lambda)\sum_{n=1}^{k-1} \lambda^{n-1} G_{t:t+n} + \lambda^{k-1} G_{t:t+k} \qquad \text{(definition)}
\end{aligned}
$$

Exercise 12.5 Several times in this book (often in exercises) we have established that returns can be written as sums of TD errors if the value function is held constant. Why is (12.10) another instance of this? Prove (12.10). □

12.4 Redoing Updates: Online λ-return Algorithm

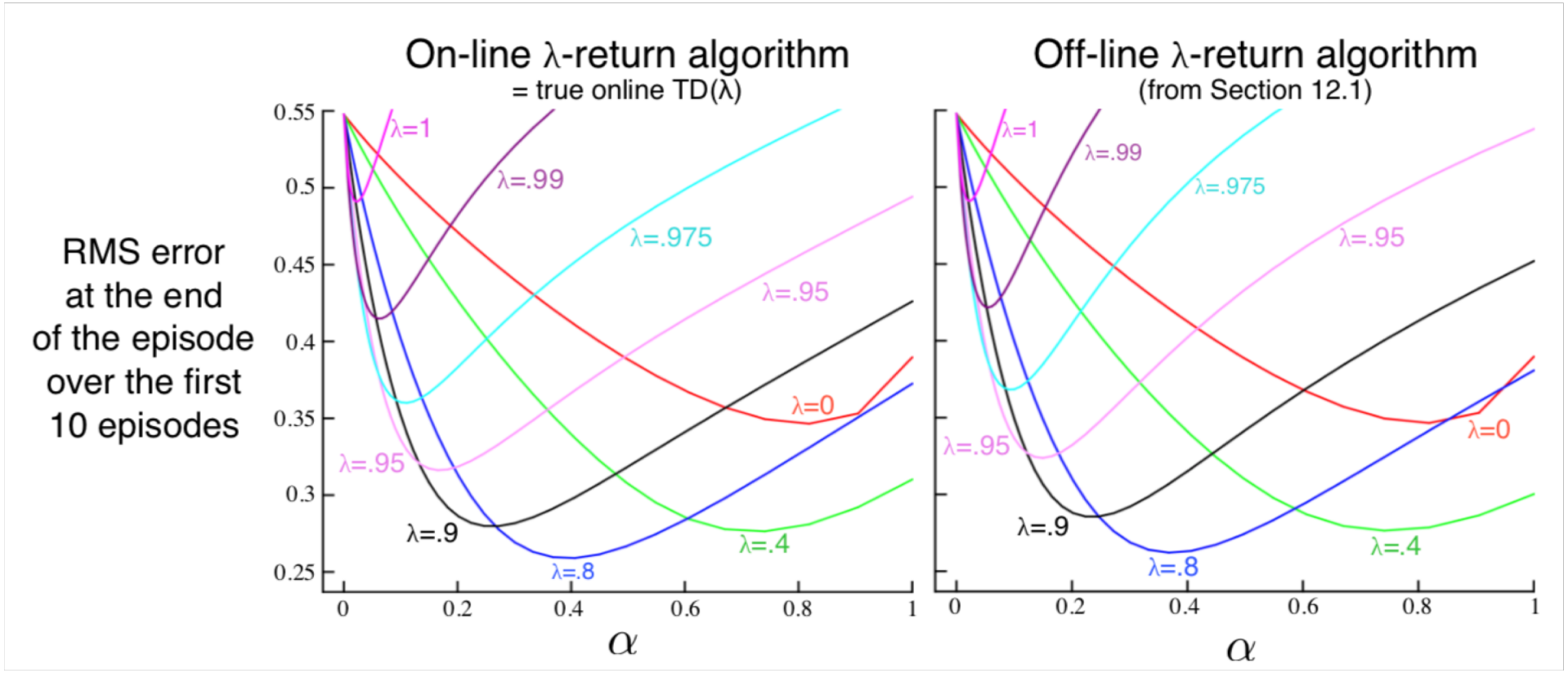
- Episode를 처음부터($t = 0$) 진행한다고 생각해 보자.
- 첫 관측(R_1)을 마치면, $G_{0:1}^{\lambda} = R_1 + \gamma\hat{v}(S_1, \mathbf{w}_0)$을 계산하여 $\mathbf{w}_1$을 계산
- 다음 관측(R_2)에서는 $G_{0:2}^{\lambda} = (1-\lambda)G_{0:1}^{\lambda} + \lambda G_{0:2}^{\lambda}$을 계산할 수 있음
- 하지만 $G_{0:2}^{\lambda}$보다는 $G_{1:2}^{\lambda}$이 더 정확함
- 두 번째 관측에서는 두 번의 update가 가능!

12.4 Redoing Updates: Online λ-return Algorithm

$$
\begin{aligned}
h=1: \quad & \mathbf{w}_1^1 \doteq \mathbf{w}_0^1 + \alpha \left[G_{0:1}^{\lambda} - \hat{v}(S_0,\mathbf{w}_0^1)\right] \nabla \hat{v}(S_0,\mathbf{w}_0^1), \\
\\
h=2: \quad & \mathbf{w}_1^2 \doteq \mathbf{w}_0^2 + \alpha \left[G_{0:2}^{\lambda} - \hat{v}(S_0,\mathbf{w}_0^2)\right] \nabla \hat{v}(S_0,\mathbf{w}_0^2), \\
& \mathbf{w}_2^2 \doteq \mathbf{w}_1^2 + \alpha \left[G_{1:2}^{\lambda} - \hat{v}(S_1,\mathbf{w}_1^2)\right] \nabla \hat{v}(S_1,\mathbf{w}_1^2), \\
\\
h=3: \quad & \mathbf{w}_1^3 \doteq \mathbf{w}_0^3 + \alpha \left[G_{0:3}^{\lambda} - \hat{v}(S_0,\mathbf{w}_0^3)\right] \nabla \hat{v}(S_0,\mathbf{w}_0^3), \\
& \mathbf{w}_2^3 \doteq \mathbf{w}_1^3 + \alpha \left[G_{1:3}^{\lambda} - \hat{v}(S_1,\mathbf{w}_1^3)\right] \nabla \hat{v}(S_1,\mathbf{w}_1^3), \\
& \mathbf{w}_3^3 \doteq \mathbf{w}_2^3 + \alpha \left[G_{2:3}^{\lambda} - \hat{v}(S_2,\mathbf{w}_2^3)\right] \nabla \hat{v}(S_2,\mathbf{w}_2^3).
\end{aligned}
$$

$$
\mathbf{w}_{t+1}^h \doteq \mathbf{w}_t^h + \alpha \left[G_{t:h}^{\lambda} - \hat{v}(S_t,\mathbf{w}_t^h)\right] \nabla \hat{v}(S_t,\mathbf{w}_t^h), \quad 0 \le t < h \le T.
$$

12.4 Redoing Updates: Online λ-return Algorithm



Larger number of informative updates

→ Better than offline algorithm!

12.5 True Online TD(λ)

- Online TD(λ)가 update해야 할 대상들 :

$$
\begin{array}{cccccc}
\mathbf{w}_0^0 & & & & & \\
\mathbf{w}_0^1 & \mathbf{w}_1^1 & & & & \\
\mathbf{w}_0^2 & \mathbf{w}_1^2 & \mathbf{w}_2^2 & & & \\
\mathbf{w}_0^3 & \mathbf{w}_1^3 & \mathbf{w}_2^3 & \mathbf{w}_3^3 & & \\
\vdots & \vdots & \vdots & \vdots & \ddots & \\
\mathbf{w}_0^T & \mathbf{w}_1^T & \mathbf{w}_2^T & \mathbf{w}_3^T & \cdots & \mathbf{w}_T^T
\end{array}
$$

- 계산량이 너무 많다!

12.5 True Online TD(λ)

- True online TD(λ) update rule ($\hat{v}(s, \mathbf{w}) = \mathbf{w}^{\mathrm{T}}\mathbf{x}(s)$, $\mathbf{x}_t = \mathbf{x}(S_t)$)

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\delta_t \mathbf{z}_t + \alpha\left(\mathbf{w}_t^\top \mathbf{x}_t - \mathbf{w}_{t-1}^\top \mathbf{x}_t\right)(\mathbf{z}_t - \mathbf{x}_t),$$

$$\mathbf{z}_t \doteq \gamma\lambda \mathbf{z}_{t-1} + \left(1 - \alpha\gamma\lambda \mathbf{z}_{t-1}^\top \mathbf{x}_t\right)\mathbf{x}_t. \qquad (12.11)$$

- 계산량이 50% 줄어듦

12.5 True Online TD(λ)

- Proof is in here(Appendix B)...

Journal of Machine Learning Research 17 (2016) 1-40 Submitted 11/15; Revised 7/16; Published 8/16

True Online Temporal-Difference Learning

Harm van Seijen[†‡] HARM.VANSEIJEN@MALUUBA.COM
A. Rupam Mahmood[†] ASHIQUE@UALBERTA.CA
Patrick M. Pilarski[†] PATRICK.PILARSKI@UALBERTA.CA
Marlos C. Machado[†] MACHADO@UALBERTA.CA
Richard S. Sutton[†] SUTTON@CS.UALBERTA.CA

12.5 True Online TD(λ)

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\delta_t \mathbf{z}_t + \alpha\left(\mathbf{w}_t^\top \mathbf{x}_t - \mathbf{w}_{t-1}^\top \mathbf{x}_t\right)(\mathbf{z}_t - \mathbf{x}_t),$$

$$\mathbf{z}_t \doteq \gamma\lambda \mathbf{z}_{t-1} + \left(1 - \alpha\gamma\lambda \mathbf{z}_{t-1}^\top \mathbf{x}_t\right)\mathbf{x}_t. \qquad (12.11)$$

True online TD(λ) for estimating $\mathbf{w}^\top\mathbf{x} \approx v_\pi$

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

Loop for each episode:
  Initialize state and obtain initial feature vector $\mathbf{x}$
  $\mathbf{z} \leftarrow \mathbf{0}$ (a d-dimensional vector)
  $V_{old} \leftarrow 0$ (a temporary scalar variable)
  Loop for each step of episode:
  | Choose $A \sim \pi$
  | Take action A, observe R, $\mathbf{x}'$ (feature vector of the next state)
  | $V \leftarrow \mathbf{w}^\top \mathbf{x}$
  | $V' \leftarrow \mathbf{w}^\top \mathbf{x}'$
  | $\delta \leftarrow R + \gamma V' - V$
  | $\mathbf{z} \leftarrow \gamma\lambda \mathbf{z} + \left(1 - \alpha\gamma\lambda \mathbf{z}^\top \mathbf{x}\right)\mathbf{x}$
  | $\mathbf{w} \leftarrow \mathbf{w} + \alpha(\delta + V - V_{old})\mathbf{z} - \alpha(V - V_{old})\mathbf{x}$
  | $V_{old} \leftarrow V'$
  | $\mathbf{x} \leftarrow \mathbf{x}'$
  until $\mathbf{x}' = \mathbf{0}$ (signaling arrival at a terminal state)

12.5 True Online TD(λ)

Dutch trace

$$\mathbf{z}_t \doteq \gamma\lambda\mathbf{z}_{t-1} + \left(1 - \alpha\gamma\lambda\mathbf{z}_{t-1}^\top\mathbf{x}_t\right)\mathbf{x}_t. \tag{12.11}$$

Best!

Accumulating trace (TD(λ))

$$\begin{aligned}\mathbf{z}_{-1} &\doteq \mathbf{0},\\ \mathbf{z}_t &\doteq \gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{v}(S_t,\mathbf{w}_t), \quad 0 \le t \le T,\end{aligned} \tag{12.5}$$

Replacing trace (only for binary feature vectors)

$$z_{i,t} \doteq \begin{cases} 1 & \text{if } x_{i,t} = 1 \\ \gamma\lambda z_{i,t-1} & \text{otherwise.} \end{cases} \tag{12.12}$$

12.7 Sarsa(λ)

- 여느 때처럼 value function → action-value function

$$G_{t:t+n} \doteq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^{n} \hat{v}(S_{t+n}, \mathbf{w}_{t+n-1}), \quad 0 \leq t \leq T - n, \quad (12.1)$$

$$G_{t:t+n} \doteq R_{t+1} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^{n} \hat{q}(S_{t+n}, A_{t+n}, \mathbf{w}_{t+n-1}), \qquad t + n < T,$$

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \left[G_t^{\lambda} - \hat{v}(S_t, \mathbf{w}_t) \right] \nabla \hat{v}(S_t, \mathbf{w}_t), \quad t = 0, \ldots, T-1. \qquad (12.4)$$

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \left[G_t^{\lambda} - \hat{q}(S_t, A_t, \mathbf{w}_t) \right] \nabla \hat{q}(S_t, A_t, \mathbf{w}_t), \quad t = 0, \ldots, T-1, \qquad (12.15)$$

12.7 Sarsa(λ)

- update rule : $$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \delta_t \mathbf{z}_t. \tag{12.7}$$

$$\delta_t \doteq R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t). \tag{12.6}$$

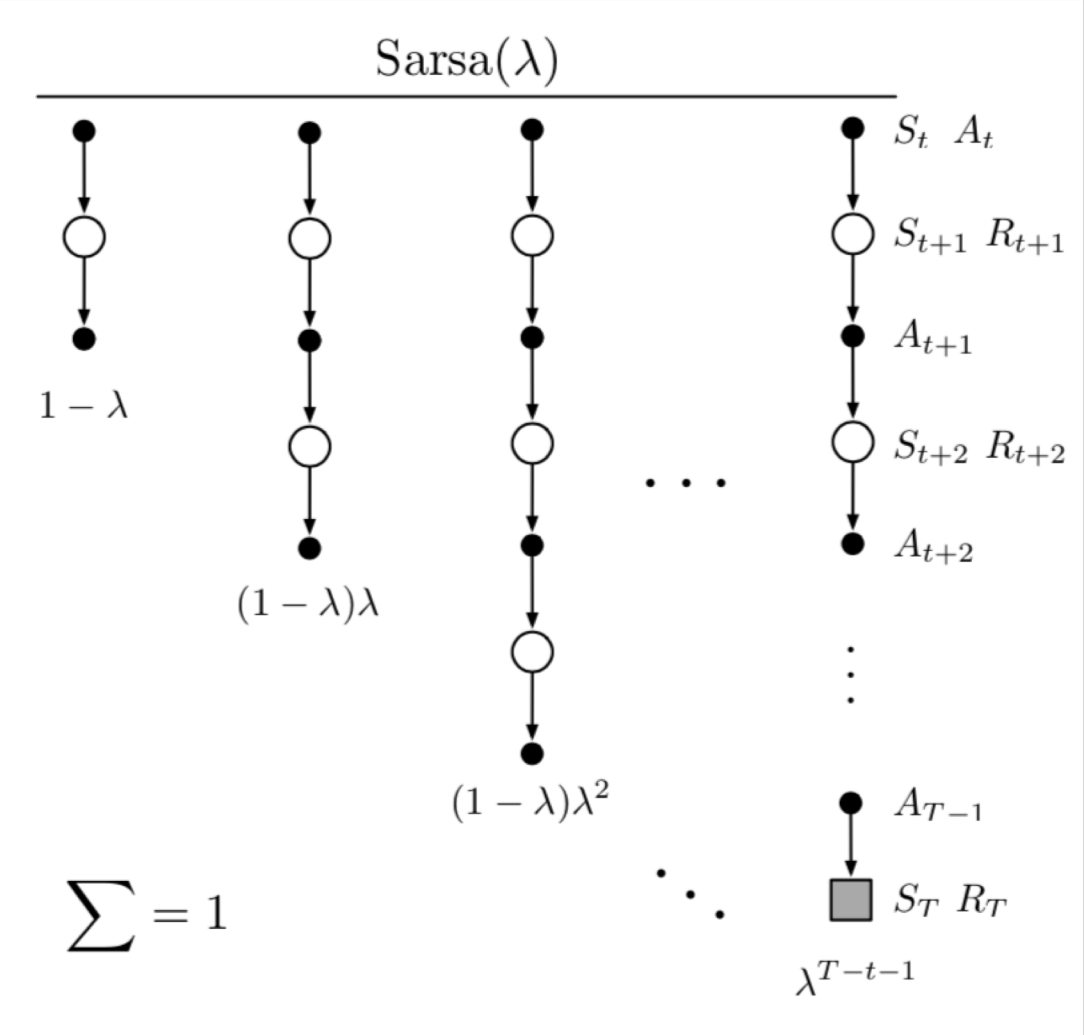
$$\delta_t \doteq R_{t+1} + \gamma \hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t), \tag{12.16}$$

$$\begin{aligned} &\mathbf{z}_{-1} \doteq \mathbf{0}, \\ &\mathbf{z}_t \doteq \gamma \lambda \mathbf{z}_{t-1} + \nabla \hat{v}(S_t, \mathbf{w}_t), \quad 0 \leq t \leq T, \end{aligned} \tag{12.5}$$

$$\begin{aligned} &\mathbf{z}_{-1} \doteq \mathbf{0}, \\ &\mathbf{z}_t \doteq \gamma \lambda \mathbf{z}_{t-1} + \nabla \hat{q}(S_t, A_t, \mathbf{w}_t), \quad 0 \leq t \leq T. \end{aligned}$$

12.7 Sarsa(λ)

- Backup diagram



12.7 Sarsa(λ)

Sarsa(λ) with binary features and linear function approximation for estimating $\mathbf{w}^\top \mathbf{x} \approx q_\pi$ or q_*

Input: a function $\mathcal{F}(s,a)$ returning the set of (indices of) active features for s, a
Input: a policy π (if estimating q_π)
Algorithm parameters: step size $\alpha > 0$, trace decay rate $\lambda \in [0,1]$
Initialize: $\mathbf{w} = (w_1, \ldots, w_d)^\top \in \mathbb{R}^d$ (e.g., $\mathbf{w} = \mathbf{0}$), $\mathbf{z} = (z_1, \ldots, z_d)^\top \in \mathbb{R}^d$

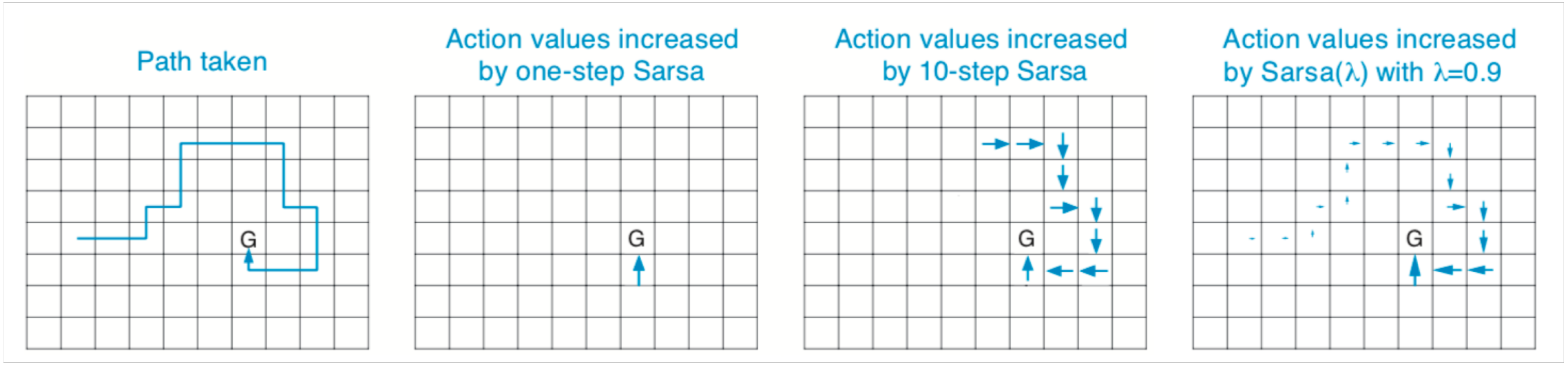
Loop for each episode:
  Initialize S
  Choose $A \sim \pi(\cdot|S)$ or ε-greedy according to $\hat{q}(S, \cdot, \mathbf{w})$
  $\mathbf{z} \leftarrow \mathbf{0}$
  Loop for each step of episode:
    Take action A, observe R, S'
    $\delta \leftarrow R$
    Loop for i in $\mathcal{F}(S, A)$:
      $\delta \leftarrow \delta - w_i$
      $z_i \leftarrow z_i + 1$ (accumulating traces)
      or $z_i \leftarrow 1$ (replacing traces)
    If S' is terminal then:
      $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
      Go to next episode
    Choose $A' \sim \pi(\cdot|S')$ or near greedily $\sim \hat{q}(S', \cdot, \mathbf{w})$
    Loop for i in $\mathcal{F}(S', A')$: $\delta \leftarrow \delta + \gamma w_i$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
    $\mathbf{z} \leftarrow \gamma\lambda\mathbf{z}$
    $S \leftarrow S'; A \leftarrow A'$

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., $\mathbf{w} = \mathbf{0}$)

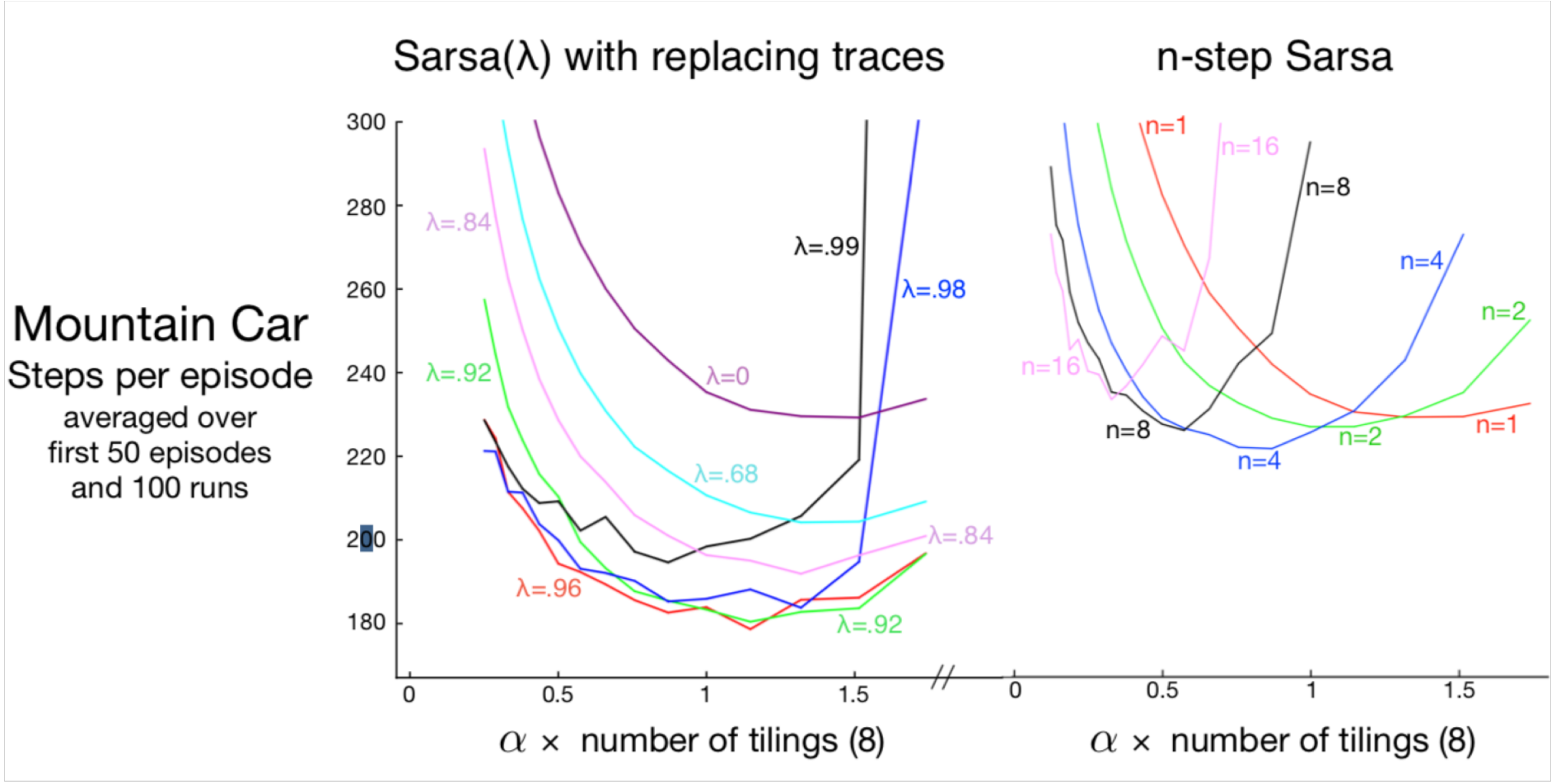
Loop for each episode:
  Initialize S
  Choose $A \sim \pi(\cdot|S)$ or near greedily from S using $\mathbf{w}$
  $\mathbf{x} \leftarrow \mathbf{x}(S, A)$
  $\mathbf{z} \leftarrow \mathbf{0}$
  $Q_{old} \leftarrow 0$
  Loop for each step of episode:
  | Take action A, observe R, S'
  | Choose $A' \sim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

12.7 Sarsa(λ) – Example 12.1

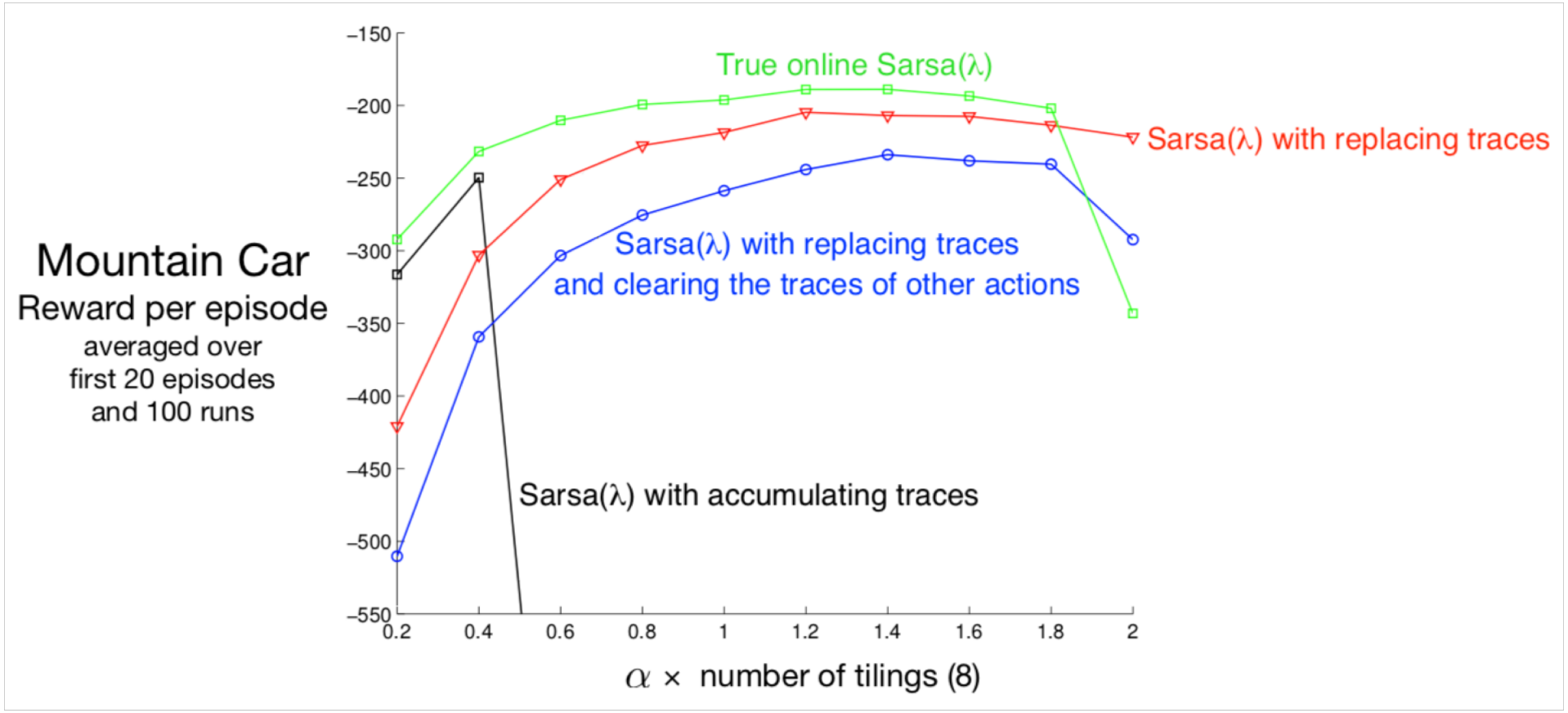


12.7 Sarsa(λ) – Example 12.2

- Mountain car example



12.7 Sarsa(λ) – Example 12.2

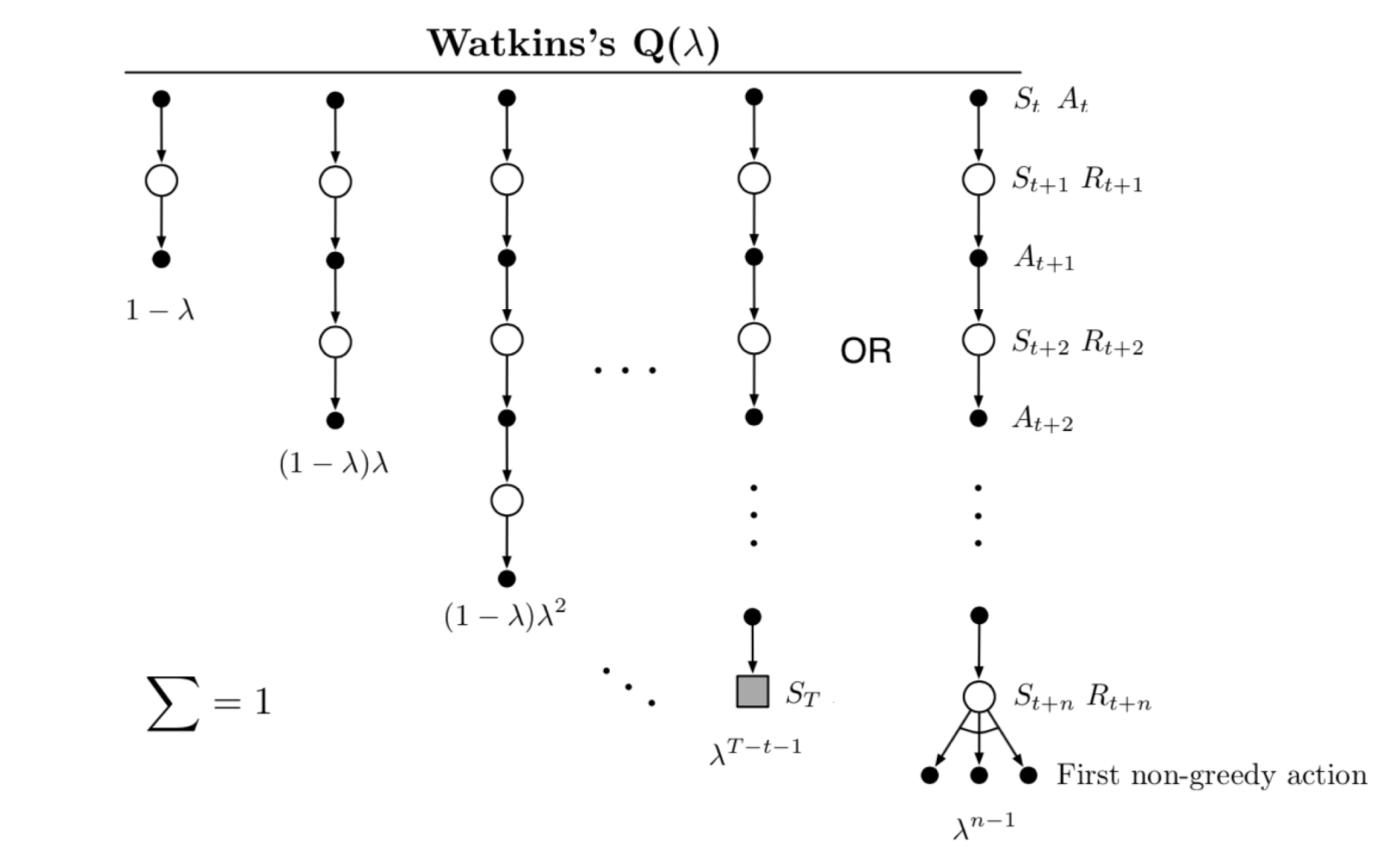


12.8 Variable λ and γ

- λ와 γ 값을 시간에 따라 다르게 (λ_t, γ_t) 줄 수도 있다!
  - $\lambda : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$
  - $\lambda_t \doteq \lambda(S_t, A_t)$
  - $\gamma : \mathcal{S} \rightarrow [0, 1]$
  - $\gamma_t \doteq \gamma(S_t)$

12.10 Watkins's Q(λ) to Tree-Backup(λ)

- Watkins's Q(λ) backup diagram



12.10 Watkins's Q(λ) to Tree-Backup(λ)

- Tree-backup(λ) update rule

$$G_t^{\lambda a} \doteq R_{t+1}+\gamma_{t+1}\Big((1-\lambda_{t+1})\bar{V}_t(S_{t+1})+\lambda_{t+1}\Big[\sum_{a\neq A_{t+1}}\pi(a|S_{t+1})\hat{q}(S_{t+1},a,\mathbf{w}_t)+\pi(A_{t+1}|S_{t+1})G_{t+1}^{\lambda a}\Big]\Big)$$

$$= R_{t+1} + \gamma_{t+1}\Big(\bar{V}_t(S_{t+1}) + \lambda_{t+1}\pi(A_{t+1}|S_{t+1})\Big(G_{t+1}^{\lambda a} - \hat{q}(S_{t+1},A_{t+1},\mathbf{w}_t)\Big)\Big)$$

- Approximation form

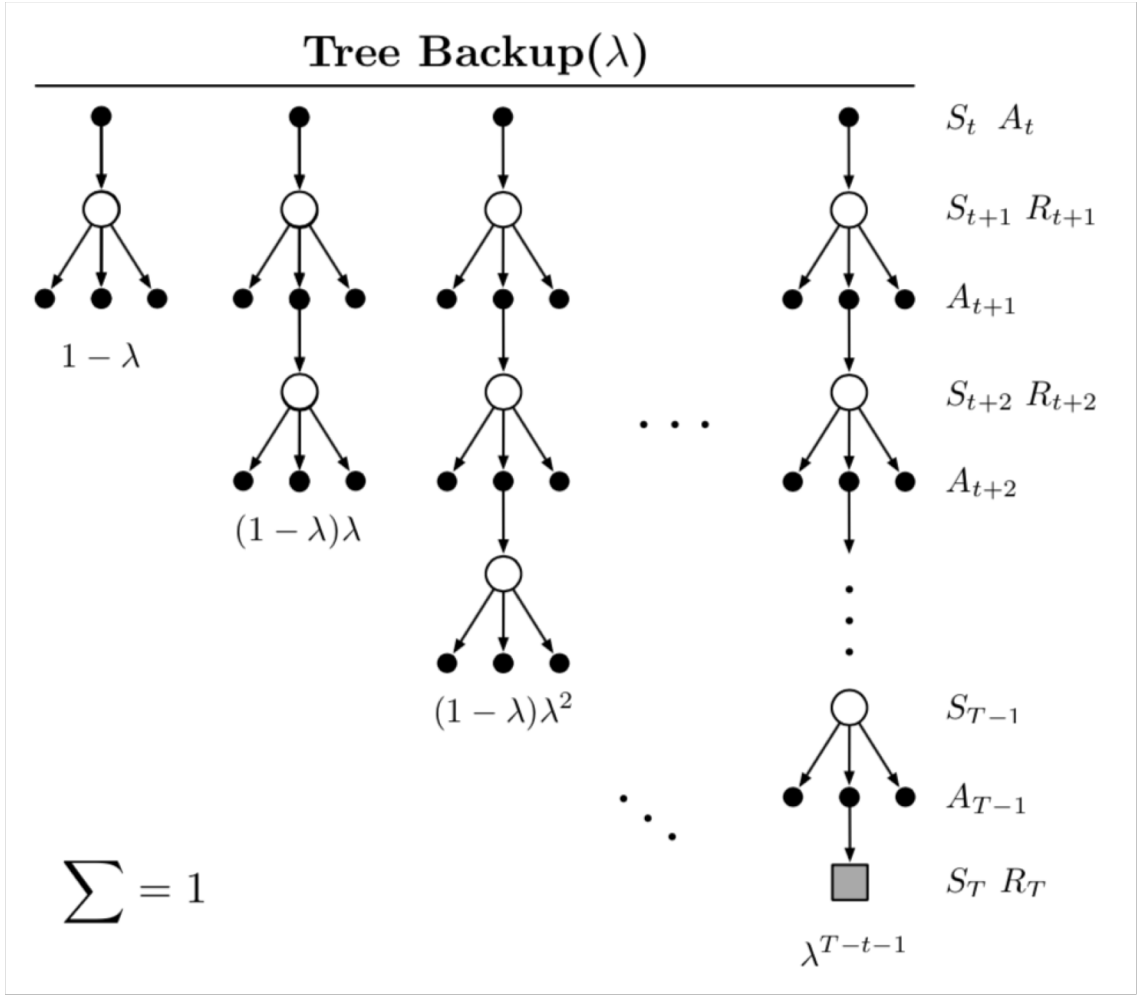
$$G_t^{\lambda a} \approx \hat{q}(S_t,A_t,\mathbf{w}_t) + \sum_{k=t}^{\infty}\delta_k^a \prod_{i=t+1}^{k}\gamma_i\lambda_i\pi(A_i|S_i),$$

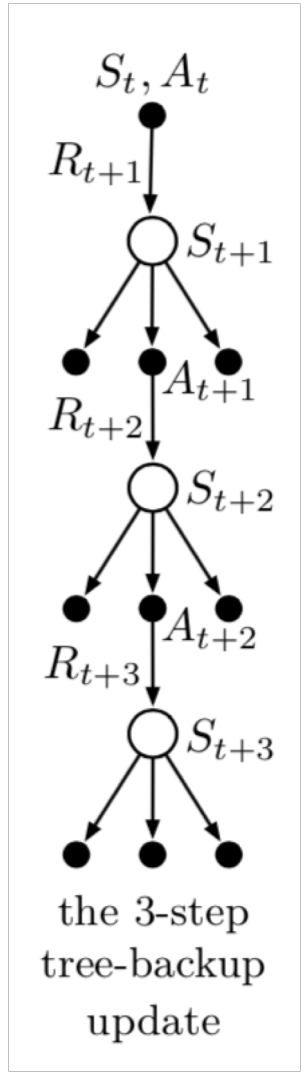
- Eligibility trace update involving the target-policy probabilities of the selected actions

$$\mathbf{z}_t \doteq \gamma_t\lambda_t\pi(A_t|S_t)\mathbf{z}_{t-1} + \nabla\hat{q}(S_t,A_t,\mathbf{w}_t).$$

12.10 Watkins's Q(λ) to Tree-Backup(λ)

- Tree-backup(λ) backup diagram





12.12 Implementation Issues

- ANN(Artificial Neural Network)으로 구현을 추천
  - value function은 매우 복잡한 형태일 것이므로 ANN이 일반적인 함수보다 정확함
- 매우 계산량이 많으므로 좋은 컴퓨터 사용을 추천

Thank you for listening!