Eligibility Traces

DR 101

2019-8-24

Resources

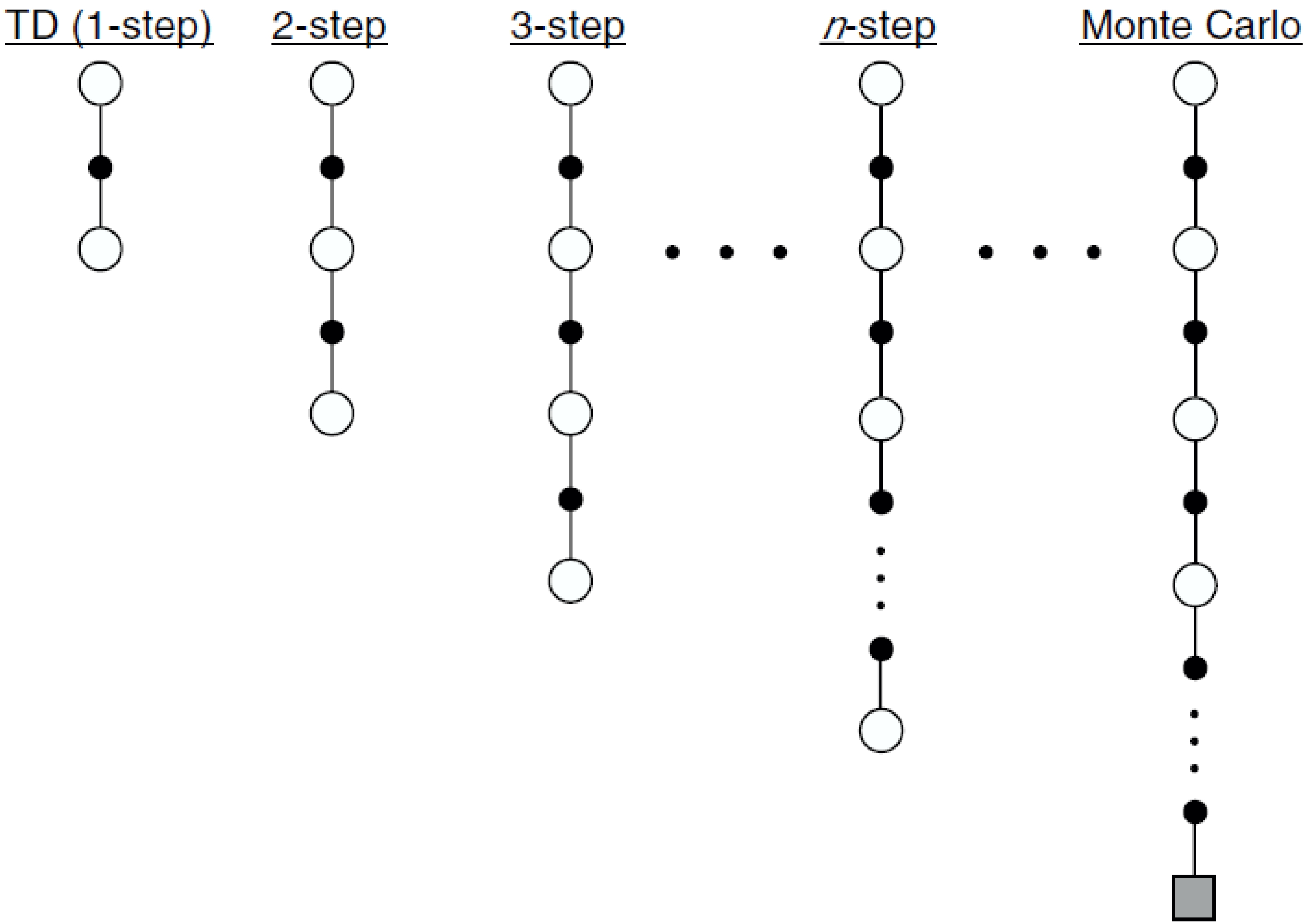
- Github
  - https://github.com/Youngsam/dr101/
- Book
  - Reinforcement Learning: An Introduction (1st edition, Chapter 7)
- Lectures
  - David Silver (lecture 4)
  - Richard Sutton (lecture 13, 21)

MC와 TD 사이의 중간길(中道)

- MC 모형의 예측은 불편향이지만 분산이 크다.
- TD 모형의 예측은 편향은 있으나 분산이 작다.
- MC와 TD의 관계:
  - N이 한 에피소드의 길이라고 할때,
  - MC=TD(N)
- 예측치의 작은 분산은 보다 빠른 학습에 있어 중요하다.
- TD(0)는 하지만 더 많은 개선이 필요하다.
- TD(n) 모형, $0 \leq n \leq N$

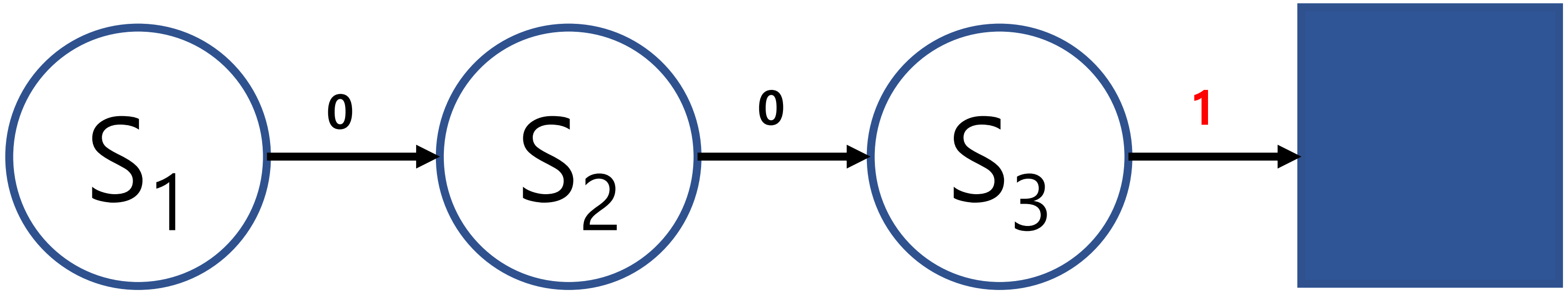
n-Step TD prediction

- Multi-Step 방법으로도 불린다.
- MC 방법의 target은 아래처럼 정해진다는 것을 떠올리자.
  - $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-t-1} R_T$
- One-Step TD의 target
  - $G_t^{(1)} = R_{t+1} + \gamma V_t(S_{t+1})$
- Two-Step TD의 target
  - $G_t^{(2)} = R_{t+1} + +\gamma R_{t+2} + \gamma^2 V_t(S_{t+1})$
- n-Step TD의 target
  - $G_t^{(n)} = R_{t+1} + +\gamma R_{t+2} + \cdots + \gamma^n V_t(S_{t+n})$
- $V(S_t) \leftarrow V(S_t) + \alpha(G_t^{(n)} - V(S_t))$



TD (1-step)
2-step
3-step
n-step
Monte Carlo

예제: TD(0)=one-step TD



초기화: $v(s_1) = v(s_2) = v(s_3) = 0,\ \alpha = \gamma = 1$

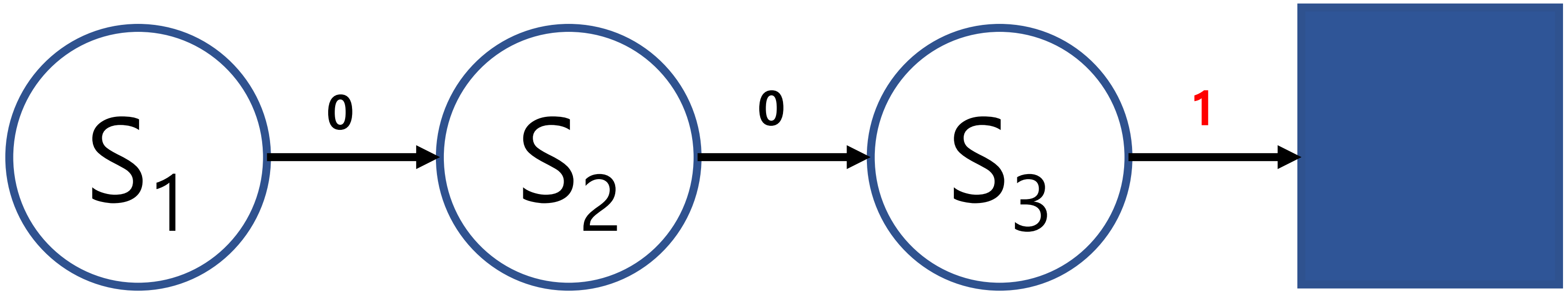
1-st 계산: $v(s_1) = v(s_1) + [r_2 + v(s_2) - v(s_1)] = 0 + (0 + 0 - 0) = 0$

2-nd 계산: $v(s_2) = v(s_2) + [r_3 + v(s_3) - v(s_2)] = 0 + (0 + 0 - 0) = 0$

3-rd 계산: $v(s_3) = v(s_3) + [r_4 - v(s_3)] = 0 + (1 - 0) = 1$

여기서 보면 가치함수가 0이 아닌 값을 갖는데 3단계가 걸렸다.

예제: two-step TD



초기화: $v(s_1) = v(s_2) = v(s_3) = 0,\ \alpha = \gamma = 1$

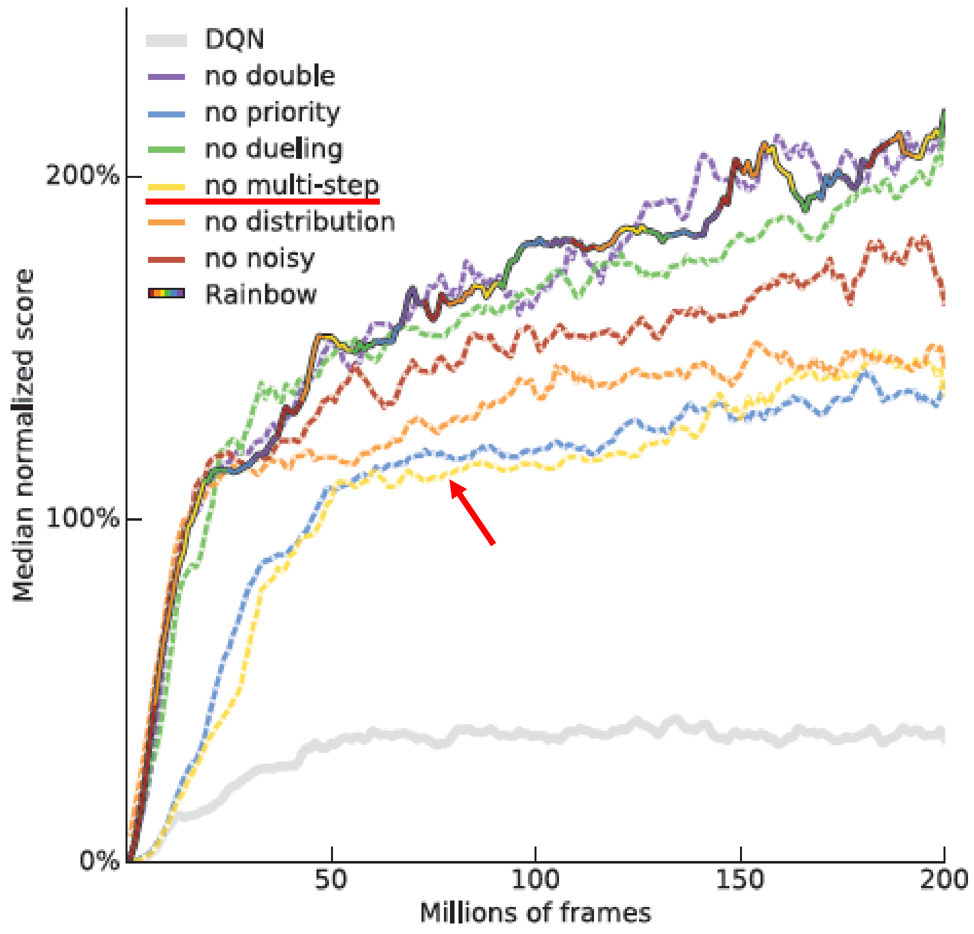
1-st 계산: $v(s_1) = v(s_1) + [r_2 + r_3 + v(s_3) - v(s_1)] = 0 + (0 + 0 + 0 - 0) = 0$

2-nd 계산: $v(s_2) = v(s_2) + [r_3 + r_4 - v(s_2)] = 0 + (0 + 1 - 0) = 1$

3-rd 계산: $v(s_3) = v(s_3) + [r_4 - v(s_3)] = 0 + (1 - 0) = 1$

여기서는 가치함수가 0이 아닌 값을 갖는데 2단계가 걸린다.

n-step TD는 학습을 빠르게 한다.



- DQN Rainbow (Hessel & Moodayil)에서는 기존에 알려진 여러 DQN 성능을 향상시키는 방법들을 선택적으로 제거하면서 어떤 방법이 가장 효과가 큰 지를 검증하였다.

Error-reduction property

- n-step return은 또한 예측 오차를 감소시켜 준다.

$$\underbrace{\max_s \left| \mathbb{E}_\pi \left[G_t^{(n)} \middle| S_t = s \right] - v_\pi(s) \right|}_{\text{n-step을 이용했을 때 최대 오차}} \leq \underbrace{\gamma^n \max_s |V(s) - v_\pi(s)|}_{\text{참된 가치함수를 통해 얻은 최대오차를 감쇄시킨 결과}}$$

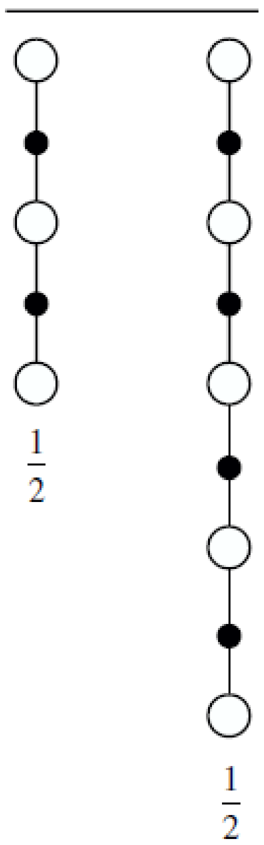
- 이 속성은 n-step 기대이익의 수렴성을 보장한다.

n-step 방법의 문제점

- 주어진 환경에 따라 적용하기가 쉽지 않다.
  - 적어도 n의 크기는 N보다는 작아야 한다.
  - 하지만 목표로 하는 환경이 제각각이라면 그 환경마다 N을 미리 파악해야만 한다.
- MC 방법과 유사하게 n-step 방법은 on-line 방법, 즉 실시간적인 방법은 아니다.
  - 만일 n이 길다면, 각 단계마다 n-step 계산이 끝나길 기다려야 한다.

n-step return을 평균낼 수 없을까?

- 한번 가치함수를 업데이트할 때, $G_t^{(n)}$를 여러 개를 놓고 그것을 평균내면 더 좋지 않을까?
- 2단계 return과 4단계 returnd의 결과를 평균을 구하는 경우를 생각해보자.
- $G_t^{avg} = \frac{1}{2} G_t^{(2)} + \frac{1}{2} G_t^{(4)}$

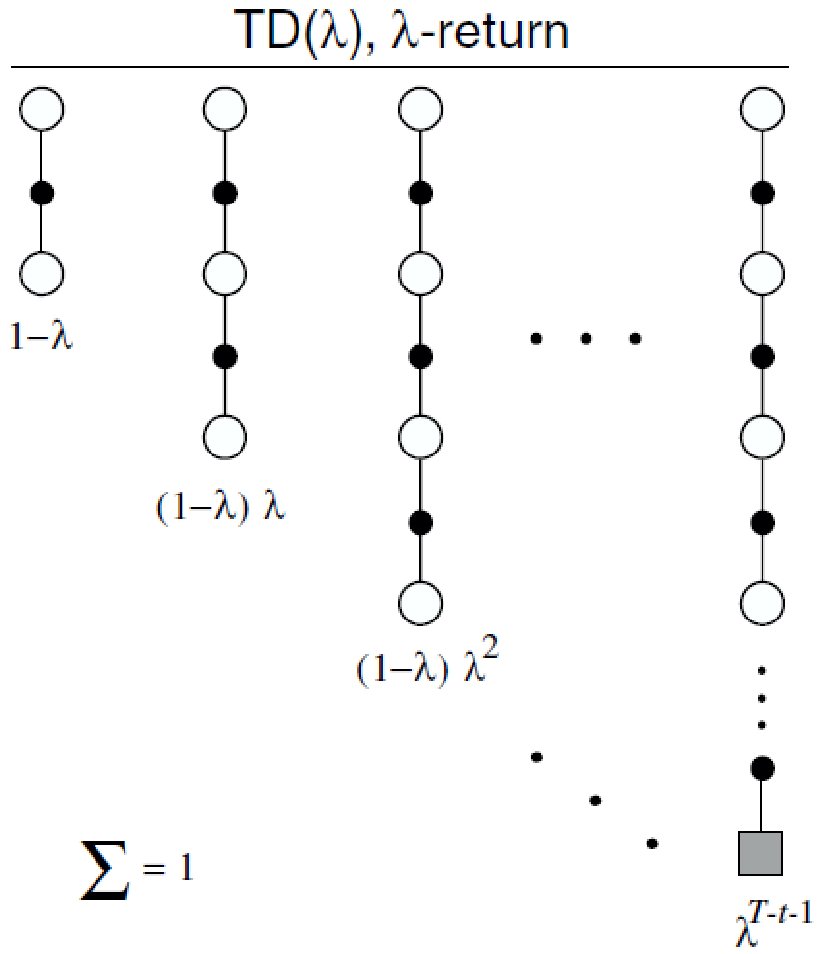


Lambda-return

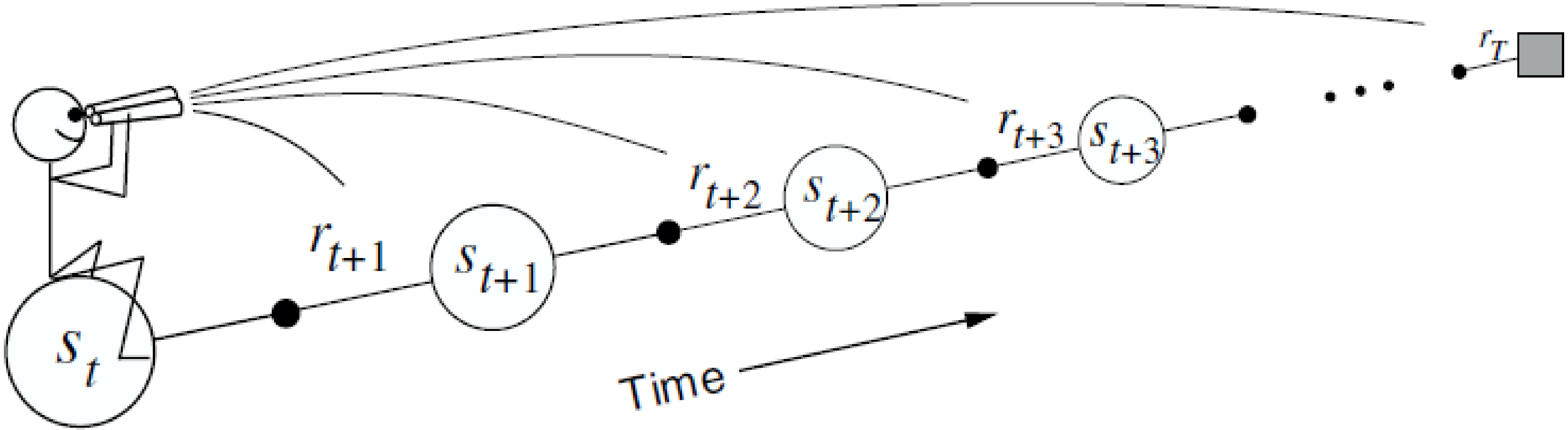
- λ-return 개념은 n-step averaging 개념을 더 세련화한 것이다.$(0 \leq \lambda \leq 1)$

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

- 이 개념을 통해 모든 n-step target을 평균을 내서 활용한다.
- 다만 각 n-step return을 λ^{n-1}만큼 감쇄시킨다.
- 이것을 보상감쇄(reward discount)와 혼동하면 안 된다.
- 보상감쇄(γ)는 이미 각 G_t안에 반영되어 있다.



Lambda-return is a forward-view



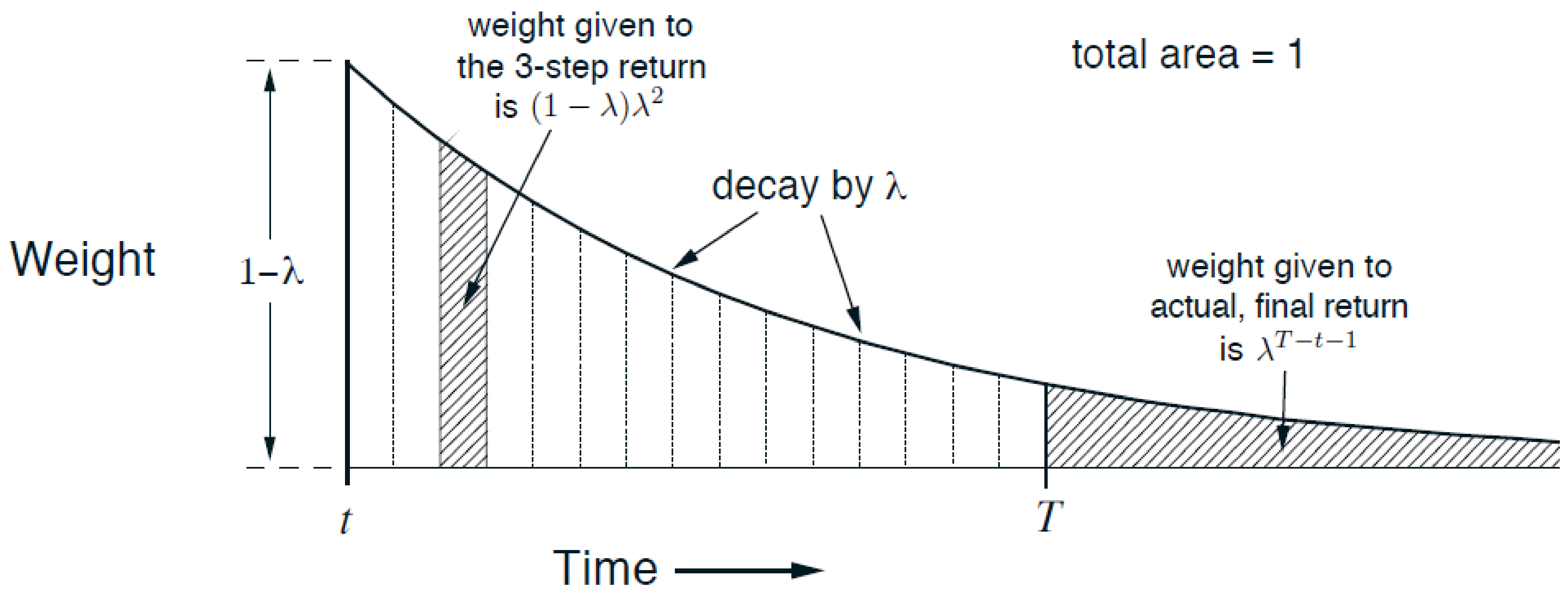
λ(0)와 λ(1)의 차이

- Lambda-return의 원래 수식

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

- 위 수식은 아래 수식으로 다시 쓸 수 있다.

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



$$G_t^{\lambda} = \underbrace{(1-\lambda)\sum_{n=1}^{T-t-1}\lambda^{n-1}G_t^{(n)}}_{\text{Until termination}} + \underbrace{\lambda^{T-t-1}G_t}_{\text{After termination}}$$

λ(0)은 TD(0)이다

$$G_t^{\lambda} = (1 - 0) \sum_{n=1}^{T-t-1} 0^{n-1} G_t^{(n)} + 0^{T-t-1} G_t$$

$$G_t^{\lambda} = \sum_{n=1}^{T-t-1} 0^{n-1} G_t^{(n)} = 0^0 G_t^{(1)} + 0^1 G_t^{(2)} + \cdots = G_t^{(1)}$$

λ(1)은 MC 방법이다

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

$$G_t^{\lambda} = 0 \sum_{n=1}^{T-t-1} 1^{n-1} G_t^{(n)} + 1^{T-t-1} G_t = 1^{T-t-1} G_t = G_t$$

Lambda의 의미

- 람다라고 하는 파라미터를 이용한 람다리턴이라는 도구를 통해 TD(0)와 MC, 즉 TD(N)을 하나의 모형으로 통합할 수 있다.
- 이제 n-step return의 n 대신 람다라는 파라미터를 통해 다양한 환경에서도 동일한 스케일의 지표를 사용할 수 있다.
- 따라서 람다를 0에 가깝게 하면, 바로 다음 상태에서 주어지는 G_t만을 현재 상태에 대한 가치함수의 갱신에 이용한다는 뜻이 된다.
- 반면에 람다를 1에 가깝게 하면 에피소드의 마지막 단계에서 주어지는 G_t 도 적극적으로 현재 상태의 가치함수에 반영하겠다는 뜻이 된다.

Lambda-return 방법의 부족한 점

- 람다-리턴을 통한 방법도 여전히 실제로 적용하기엔 불편한 점이 있다.
- 불편함의 원인은 이 방법이 forward-view에 의존한다는 점에 있다.
- 결국 실제 학습을 진행하려면 환경이 람다-리턴을 제공하는 시기까지 에이전트는 대기해야 한다는 문제점이 있다.
- 그렇다면 이것은 실시간 예측을 제공하지는 못하는 학습체계라는 것을 뜻한다.

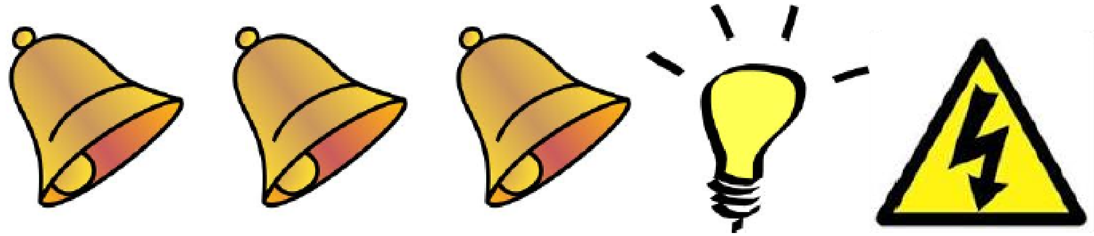
Eligibility Traces

- Eligibility traces (적격 흔적도)라는 장치를 통해 Sutton (1984, 1988)은 TD 모형을 backward-view 방식으로 구현하게 된다.
- 이것을 TD(λ) 모형이라고 부른다.
- 여기서 람다라는 파라미터는 이전의 단순한 가중치로서의 의미가 아니라 trace-decay 파라미터, 즉 흔적-감쇄율이라는 의미를 갖게 된다.

Klopf’s Eligibility Traces

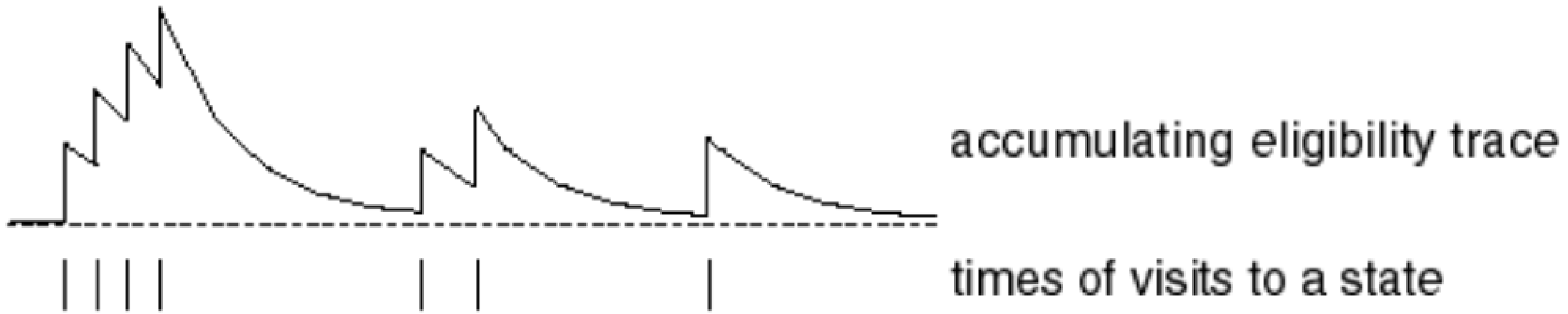
- Klopf (1972)는 뉴런의 시냅스들이 특정 조건 하에서 자극을 전달 받기에 eligible (적합해지는) 현상에 대한 이론을 제시한다.
- 그는 뉴런 끼리 신호가 전달이 된 후에 그것에 대한 기억정보가 뉴런들 사이의 시냅스에 잔존한다고 가정했다.
- 이러한 자극의 흔적에 대한 기억장치를 eligibility traces라고 부르며, Sutton도 그런 의미로 이 용어를 사용한다.
- Sutton은 이런 일시적 기억능력이 생물이 credit assignment 문제를 푸는데 유용하게 활용된다고 보았다.

Credit Assignment Problem



- Credit assignment problem: did bell or light cause shock?
- Frequency heuristic: assign credit to most frequent states
- Recency heuristic: assign credit to most recent states
- *Eligibility traces* combine both heuristics

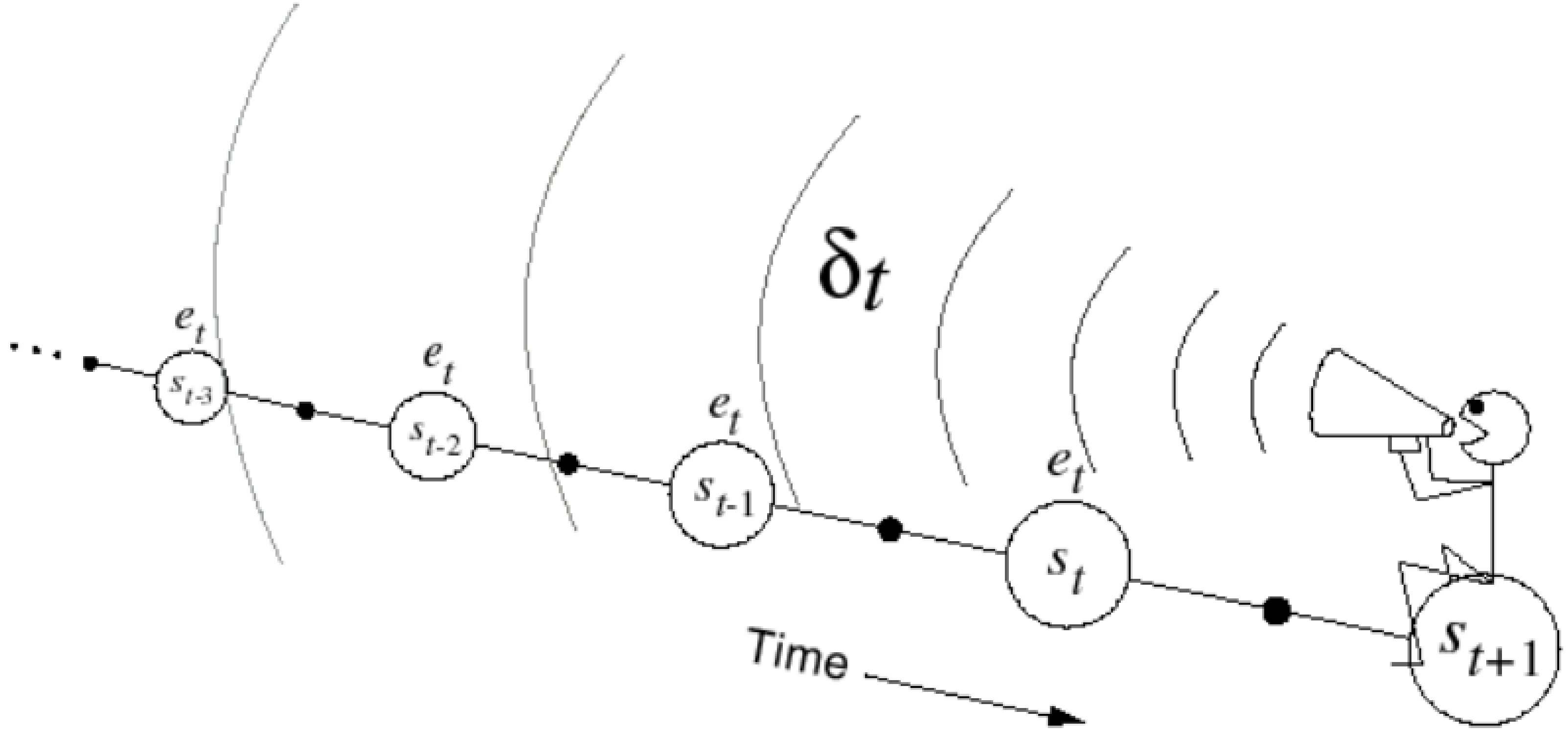
$$E_0(s) = 0$$
$$E_t(s) = \gamma\lambda E_{t-1}(s) + \mathbf{1}(S_t = s)$$



Backward View of TD(λ)

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

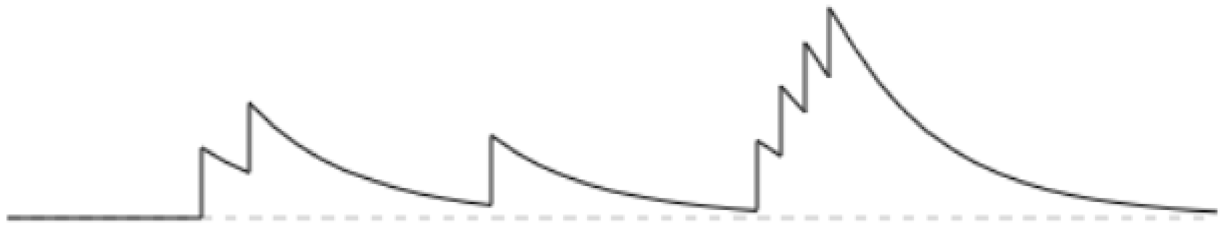
$$V(s) \leftarrow V(s) + \alpha \delta_t E_t(s)$$



Sutton’s Eligibility Trace

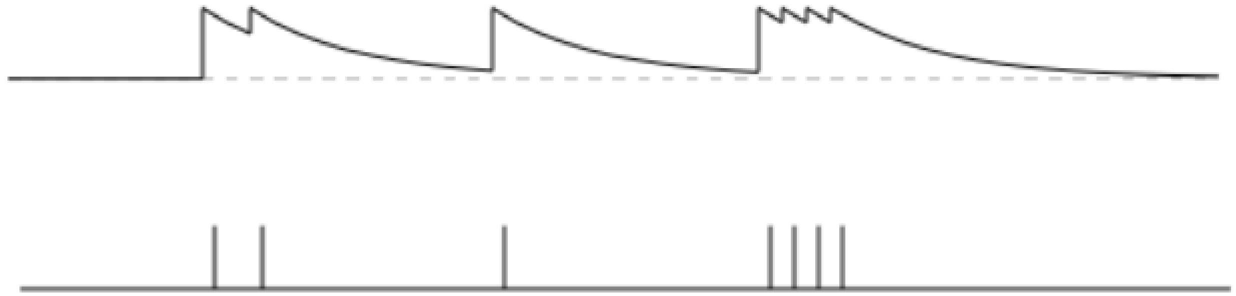
- Accumulating trace

$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1}(s) & \text{if } s \neq s_t \\ \gamma\lambda e_{t-1}(s) + 1 & \text{if } s = s_t \end{cases}$$



- Replacing trace

$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1}(s) & \text{if } s \neq s_t \\ 1 & \text{if } s = s_t \end{cases}$$



On-line Tabular TD(λ)

Initialize $V(s)$ arbitrarily and $e(s) = 0$, for all $s \in \mathcal{S}$
Repeat (for each episode):
    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'
        $\delta \leftarrow r + \gamma V(s') - V(s)$
        $e(s) \leftarrow e(s) + 1$
        For all s:
            $V(s) \leftarrow V(s) + \alpha\delta e(s)$
            $e(s) \leftarrow \gamma\lambda e(s)$
        $s \leftarrow s'$
    until s is terminal

TD(λ), $\lambda = 0$

- TD-람다 알고리즘을 이용한 학습 중:
  - $S_t = s$
  - $\delta \leftarrow R_{t+1} + \gamma V(S_{t+1}) - V(s)$
  - $e(s) \leftarrow e(s) + 1 = 0 + 1$
  - $V(s) \leftarrow V(s) + \alpha\delta_t e_t(s) = V(s) + \alpha\delta_t$

- 위에서 보듯이, 람다가 0이면 TD 람다 모형은 TD(0)와 같아진다.

TD(λ), $\lambda = 1$

- TD-람다 알고리즘을 이용한 학습 중:
  - $\delta \leftarrow R_{t+1} + \gamma V(S_{t+1}) - V(s)$
  - $e(s) \leftarrow \gamma e(s) + 1$
    - (Inner loop)
    - $V(s) \leftarrow V(s) + \alpha \delta_t e_t(s)$
    - $e(s) \leftarrow \gamma \lambda e(s) = \gamma^k$

- 위에서 보듯이, 람다가 1이면 $e(s)$은 거의 감소하지 않는다.
- 즉, 이것은 MC와 결과적으로 같다.

On-line Tabular Sarsa(λ)

Initialize $Q(s, a)$ arbitrarily and $e(s, a) = 0$, for all s, a
Repeat (for each episode):
  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)
    $\delta \leftarrow r + \gamma Q(s', a') - Q(s, a)$
    $e(s, a) \leftarrow e(s, a) + 1$
    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

On-line Tabular Q(λ)

Initialize $Q(s, a)$ arbitrarily and $e(s, a) = 0$, for all s, a
Repeat (for each episode):
    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)
        $a^* \leftarrow \arg\max_b Q(s', b)$ (if a' ties for the max, then $a^* \leftarrow a'$)
        $\delta \leftarrow r + \gamma Q(s', a^*) - Q(s, a)$
        $e(s, a) \leftarrow e(s, a) + 1$
        For all s, a:
            $Q(s, a) \leftarrow Q(s, a) + \alpha \delta e(s, a)$
            If $a' = a^*$, then $e(s, a) \leftarrow \gamma \lambda e(s, a)$
                else $e(s, a) \leftarrow 0$
        $s \leftarrow s'; a \leftarrow a'$
    until s is terminal

최종요약

- n-step TD 방법을 통해 TD(0)의 약점을 줄일 수 있으나 MC의 단점을 떠안게 된다.
- 람다-리턴을 이용한 TD 방법은 n-step TD와 MC를 forward-view 방식으로 매끄럽게 이어준다.
- Eligibility trace를 이용한 TD-람다 모형은 backward-view 방식으로 TD 모형과 MC 모형을 통합시켜준다.
- 이로써 실시간 분석이라는 TD모형의 장점을 온전하게 누릴 수 있다.
- 이론적으로 backward-view 기반 TD-람다 모형은 forward-view 방식의 TD 모형과 동치는 아니었으나, 최근에는 그것을 보장하는 TD 모형도 개발되었다.(True Online TD)