强化学习及其应用

Reinforcement Learning and Its Applications

第三章 值估计

Value Evaluation

授课人：周晓飞
zhouxiaofei@iie.ac.cn
2023-6-13

第三章 值估计

3.1 随机逼近

3.2 蒙特卡洛值估计

3.3 时序差分值估计

3.4 算法总结

第三章 值估计

3.1 随机逼近

3.2 蒙特卡洛值估计

3.3 时序差分值估计

3.4 算法总结

随机逼近

累计平均逼近 E(x)

对于随机变量 x，概率分布 P(x)未知，估计期望 E(x):

For time k:

$S \leftarrow S + x_k$;

$N \leftarrow N + 1$;

$u_k \leftarrow S/N$;

$N \to \infty, \quad u \to E(x)$

随机逼近

增量均值逼近 E(*x*)

For time k:

$N \leftarrow N+1;$

$u_k \leftarrow u_{k-1}+(1/N)\ (x_k - u_{k-1});$

相当于均值的更新：

$$\begin{aligned}\mu_k &= \frac{1}{k}\sum_{j=1}^{k} x_j \\ &= \frac{1}{k}\left(x_k + \sum_{j=1}^{k-1} x_j\right) \\ &= \frac{1}{k}\left(x_k + (k-1)\mu_{k-1}\right) \\ &= \mu_{k-1} + \frac{1}{k}\left(x_k - \mu_{k-1}\right)\end{aligned}$$

随机逼近

Robbins-Monro 逼近 E(*x*)

For time k:

$$N \leftarrow N+1;$$

$$u_k \leftarrow u_{k-1} + \alpha\,(x_k - u_{k-1});$$

相当于权重比例更新：

$$u_k \leftarrow (1-\alpha)u_{k-1} + \alpha\, x_k$$

本课程常用 Robbins-Monro 随机逼近公式。

第三章 值估计

3.1 随机逼近

3.2 蒙特卡洛值估计

3.3 时序差分值估计

3.4 算法总结

蒙特卡洛值估计

问题描述

- Goal: learn v_π from episodes of experience under policy π

$$S_1, A_1, R_2, ..., S_k \sim \pi$$

- Recall that the *return* is the total discounted reward:

$$G_t = R_{t+1} + \gamma R_{t+2} + ... + \gamma^{T-1} R_T$$

- Recall that the value function is the expected return:

$$v_\pi(s) = \mathbb{E}_\pi \left[G_t \mid S_t = s \right]$$

- Monte-Carlo policy evaluation uses *empirical mean* return instead of *expected* return

蒙特卡洛值估计

First-Visit MC Evaluation

- To evaluate state s
- The first time-step t that state s is visited in an episode,
- Increment counter $N(s) \leftarrow N(s) + 1$
- Increment total return $S(s) \leftarrow S(s) + G_t$
- Value is estimated by mean return $V(s) = S(s)/N(s)$
- By law of large numbers, $V(s) \rightarrow v_\pi(s)$ as $N(s) \rightarrow \infty$

只对 episode 的起始状态进行统计。

蒙特卡洛值估计

Every-Visit MC Evaluation

- To evaluate state s
- Every time-step t that state s is visited in an episode,
- Increment counter $N(s) \leftarrow N(s) + 1$
- Increment total return $S(s) \leftarrow S(s) + G_t$
- Value is estimated by mean return $V(s) = S(s)/N(s)$
- Again, $V(s) \rightarrow v_\pi(s)$ as $N(s) \rightarrow \infty$

episode 的每个状态进行统计。

蒙特卡洛值估计

Incremental MC Evaluation

- Update $V(s)$ incrementally after episode $S_1, A_1, R_2, ..., S_T$
- For each state S_t with return G_t

$$N(S_t) \leftarrow N(S_t) + 1$$

$$V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}(G_t - V(S_t))$$

蒙特卡洛值估计

Incremental MC Evaluation

- Update $V(s)$ incrementally after episode $S_1, A_1, R_2, ..., S_T$
- For each state S_t with return G_t

$$N(S_t) \leftarrow N(S_t) + 1$$

$$V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}(G_t - V(S_t))$$

- In non-stationary problems, it can be useful to track a running mean, i.e. forget old episodes.

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$$

第三章 值估计

3.1 随机逼近

3.2 蒙特卡洛值估计

3.3 时序差分值估计

3.4 算法总结

时序差分值估计

问题描述

采用不完整的 episodes，估计V值。

- TD methods learn directly from episodes of experience
- TD is *model-free*: no knowledge of MDP transitions / rewards
- TD learns from *incomplete* episodes, by *bootstrapping*
- TD updates a guess towards a guess

时序差分值估计

TD (0)

- Update value $V(S_t)$ toward *estimated* return $R_{t+1} + \gamma V(S_{t+1})$

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

Bellman 迭代的随机形式

- $R_{t+1} + \gamma V(S_{t+1})$ is called the *TD target*
- $\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$ is called the *TD error*

时序差分值估计

TD (0)

V.S. MC

- Incremental every-visit Monte-Carlo
  - Update value $V(S_t)$ toward *actual* return G_t

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

时序差分值估计

TD (0)

- TD can learn *before* knowing the final outcome
  - TD can learn online after every step
  - MC must wait until end of episode before return is known
- TD can learn *without* the final outcome
  - TD can learn from incomplete sequences
  - MC can only learn from complete sequences
  - TD works in continuing (non-terminating) environments
  - MC only works for episodic (terminating) environments

时序差分值估计

TD (0)

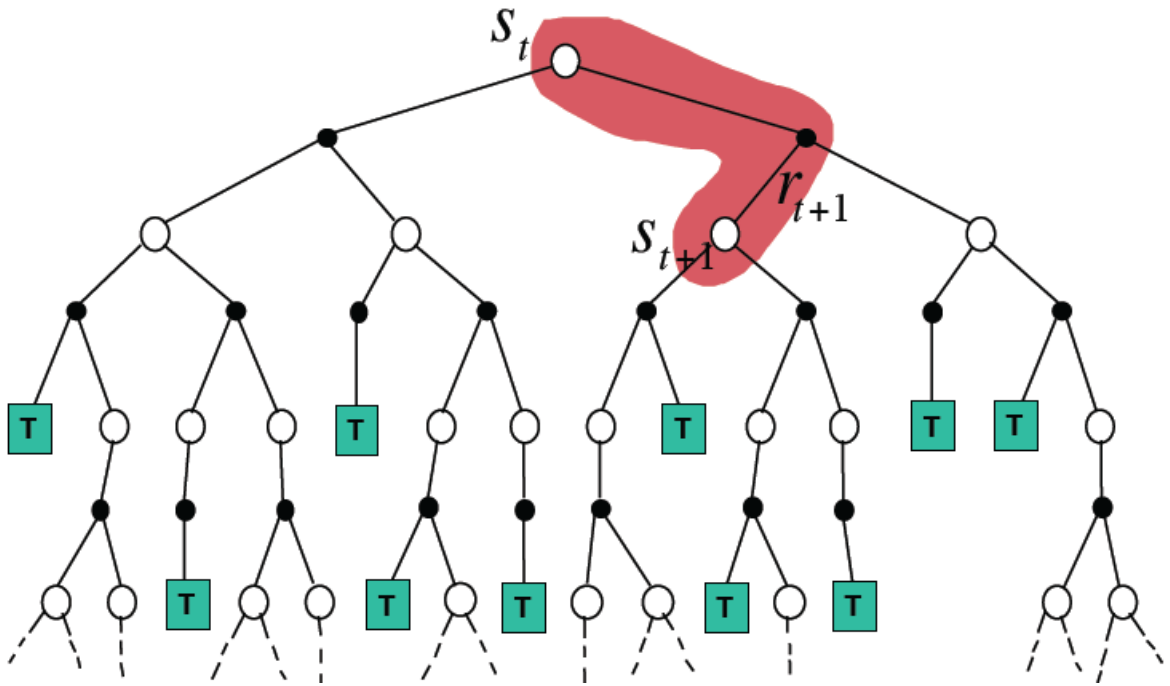
- TD exploits Markov property
  - Usually more efficient in Markov environments
- MC does not exploit Markov property
  - Usually more effective in non-Markov environments

时序差分值估计

TD (0)

TD Backup

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

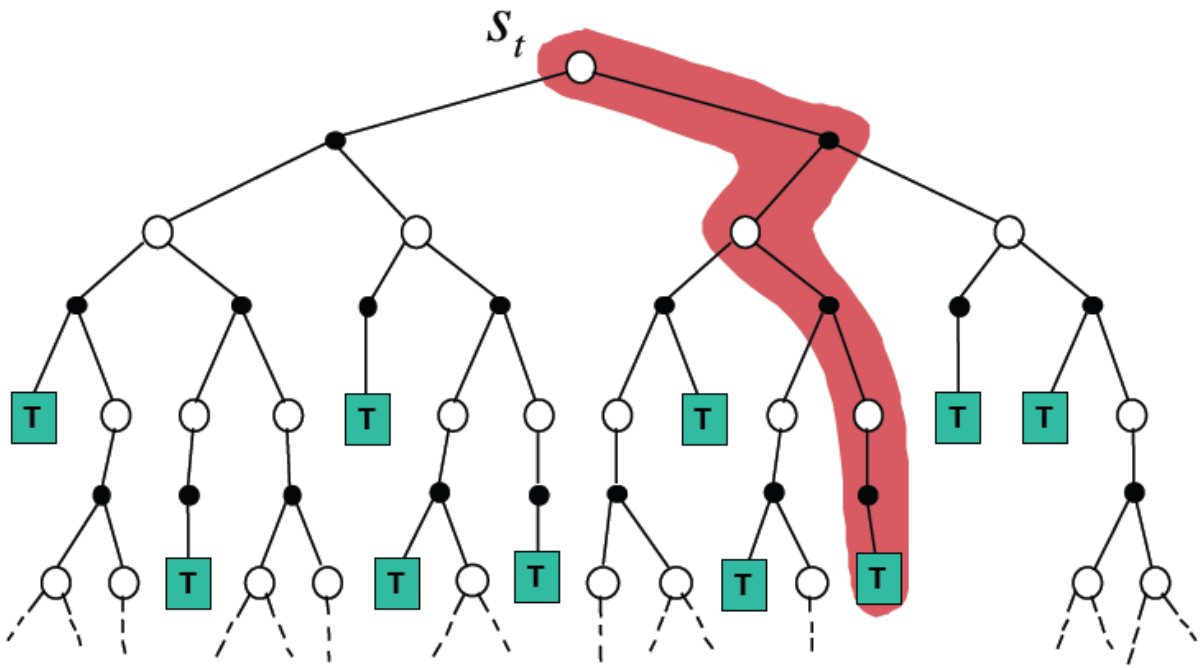


时序差分值估计

TD (0)

V.S. MC

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

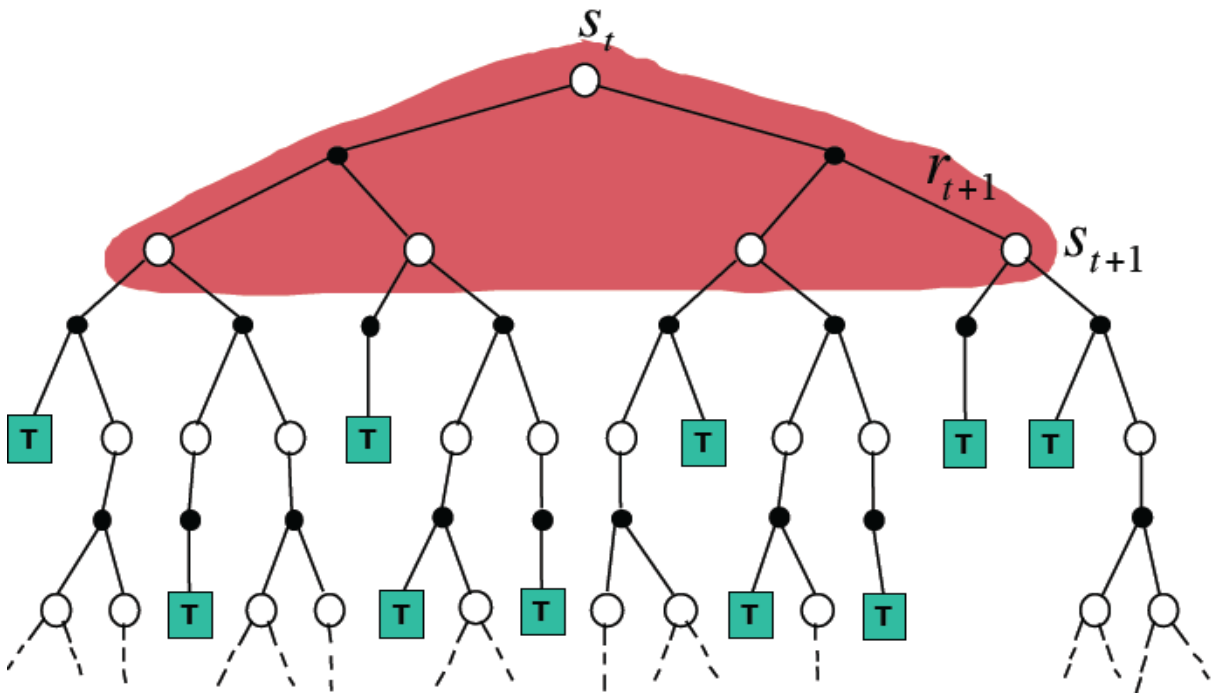


时序差分值估计

TD (0)

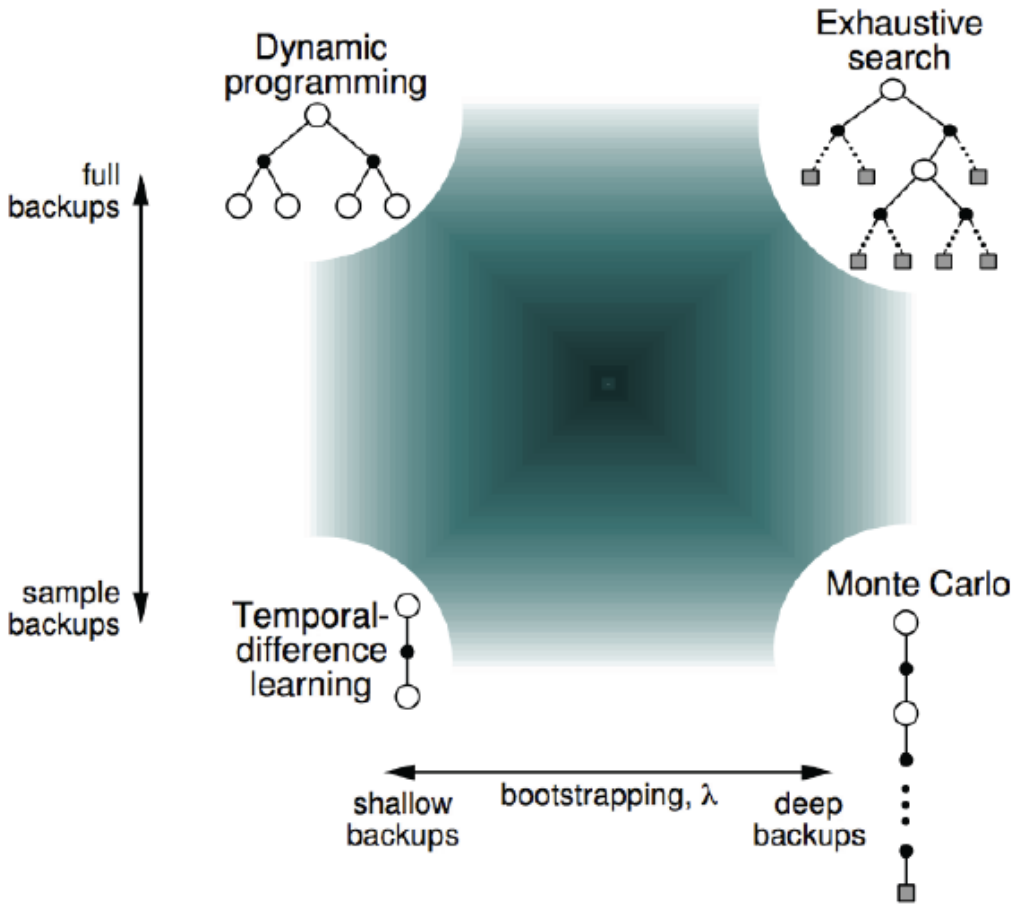
V.S. DP

$$V(S_t) \leftarrow \mathbb{E}_\pi \left[R_{t+1} + \gamma V(S_{t+1}) \right]$$



时序差分值估计

TD (0)


Dynamic programming
Exhaustive search
full backups
sample backups
Temporal-difference learning
Monte Carlo
shallow backups
bootstrapping, λ
deep backups

时序差分值估计

TD (0)

Bootstrapping and Sampling

- Bootstrapping: update involves an estimate
  - MC does not bootstrap
  - DP bootstraps
  - TD bootstraps
- Sampling: update samples an expectation
  - MC samples
  - DP does not sample
  - TD samples

时序差分值估计

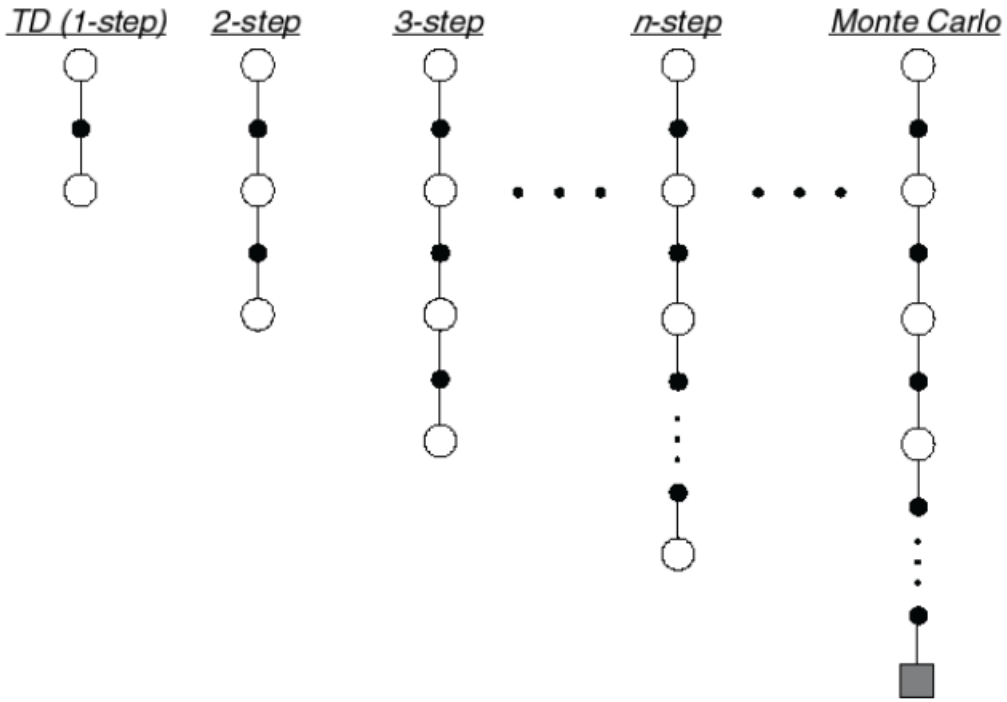
Have a break !

时序差分值估计

TD(λ)

n-step TD

- Let TD target look *n* steps into the future



时序差分值估计

TD(λ)

- 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

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

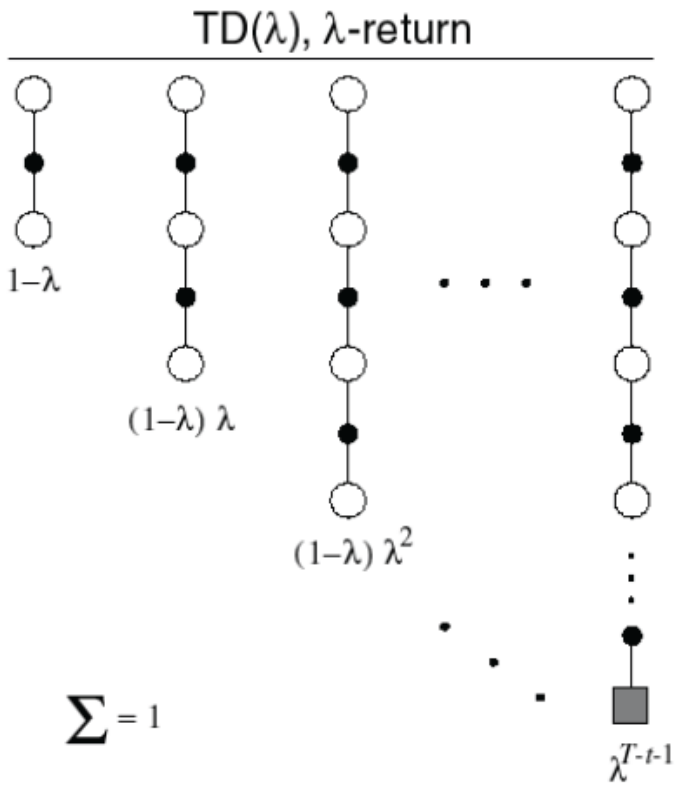
- n-step temporal-difference learning

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

时序差分值估计

TD(λ)

Forward-view TD(λ)



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

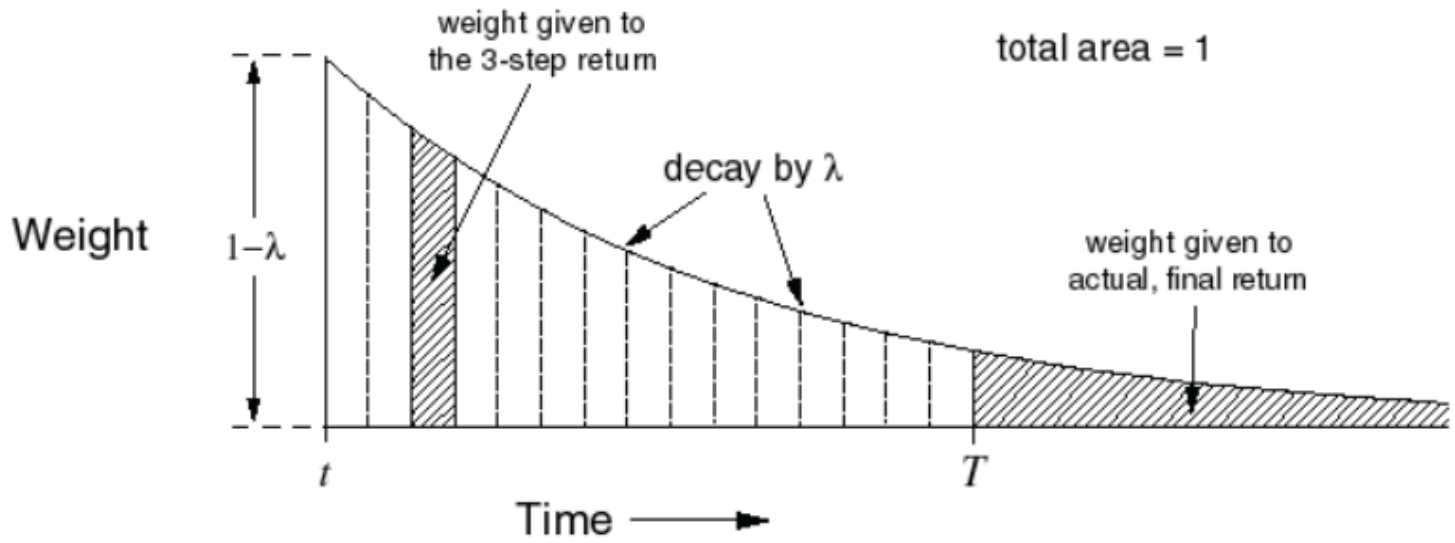
$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_t^{(n)}$$

- Forward-view TD(λ)

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

时序差分值估计

TD(λ)



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

时序差分值估计

TD(λ)

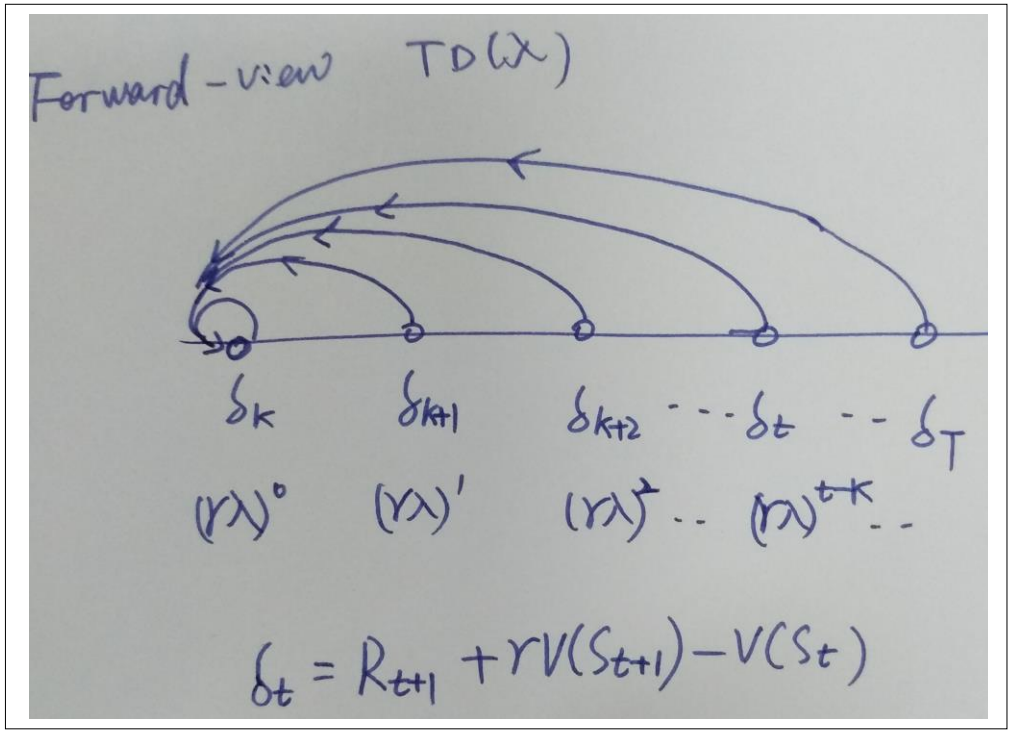
$$\alpha\left(G_k^\lambda - V(S_k)\right) = \alpha\sum_{t=k}^{T}(\gamma\lambda)^{t-k}\delta_t$$

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

(等式推导见 David 的 PPT)

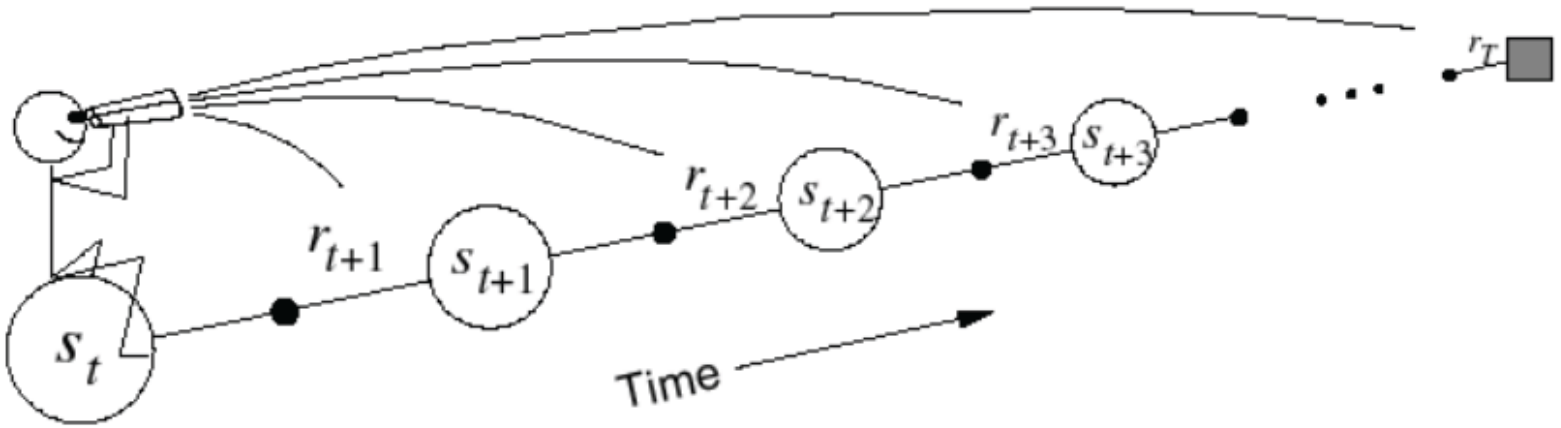
$$V(s) \leftarrow V(s) + \alpha\sum_{t=k}^{T}(\gamma\lambda)^{t-k}\delta_t$$

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



时序差分值估计

TD(λ)

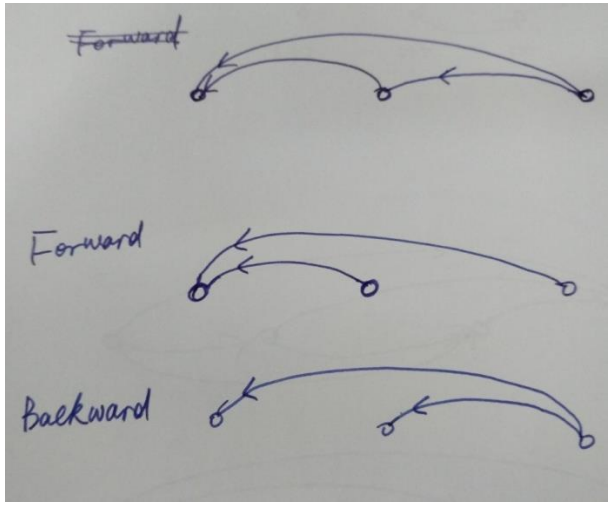


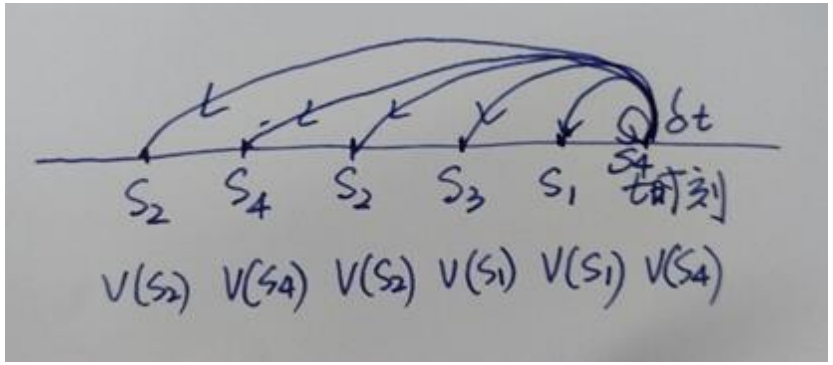
- Update value function towards the λ-return
- Forward-view looks into the future to compute G_t^λ
- Like MC, can only be computed from complete episodes

时序差分值估计

TD(λ)

Backward view TD(λ)





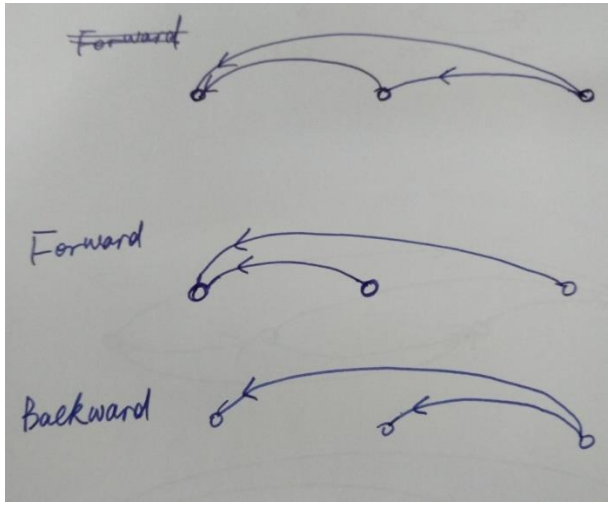
分别对之前每个时刻的状态进行修正

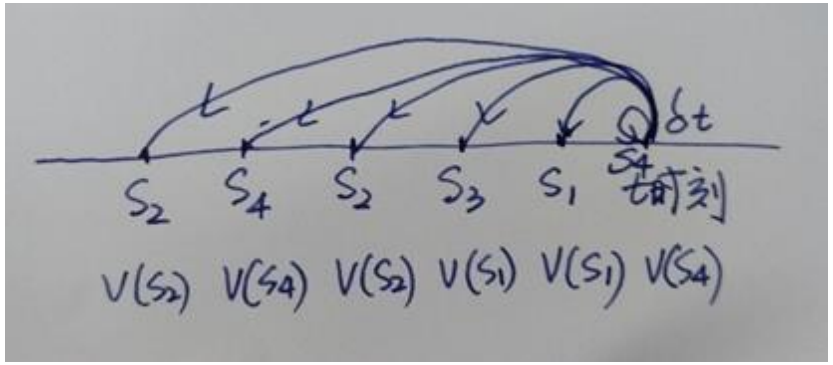
$$V(s) \leftarrow V(s) + \alpha(\gamma\lambda)^{t-k}\delta_t$$

时序差分值估计

TD(λ)

Backward view TD(λ)





分别对之前每个时刻的状态进行修正

$$V(s) \leftarrow V(s) + \alpha(\gamma\lambda)^{t-k}\delta_t$$

更好的简化办法：引入资格迹 E(S)，对每个状态出现的时刻统计，并对每个状态值修正

时序差分值估计

TD(λ)

Backward view TD(λ)

对所有状态进行更新

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

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

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

时序差分值估计

TD(λ)

Backward view TD(λ)

资格迹：每个状态的资格迹，随时间更新，记录了状态出现的位置。

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

例子：S = {S_1, S_2}； Episodes: S_1, S_2, S_2, S_1, S_1；当前时刻 t 之前 出现状态的标定：

t=0:		$E_0(S_1)=0$;	$E_0(S_2)=0$
t=1:	S_1	$E_1(S_1)=r\lambda E_0(S_1)+1=1$	$E_1(S_2)= r\lambda E_0(S_2) =0$
t=2:	S_1, S_2,	$E_2(S_1)=r\lambda E_1(S_1)=r\lambda$	$E_2(S_2)= r\lambda E_1(S_2)+1=1$
t=3:	S_1, S_2, S_2,	$E_3(S_1)=r\lambda E_2(S_1)=(r\lambda)^2$	$E_3(S_2)= r\lambda E_2(S_2)+1=(r\lambda)+1$
t=4:	S_1, S_2, S_2, S_1,	$E_4(S_1)=r\lambda E_3(S_1)+1=(r\lambda)^3+1$;	$E_4(S_2)=r\lambda E_3(S_2) =(r\lambda)^2+(r\lambda)$
t=5:	S_1, S_2, S_2, S_1, S_1	$E_5(S_1)=r\lambda E_4(S_1)+1=(r\lambda)^4+r\lambda+1$	$E_5(S_2)=r\lambda E_4(S_2) =(r\lambda)^3+(r\lambda)^2$

时序差分值估计

TD(λ)

Backward view TD(λ)

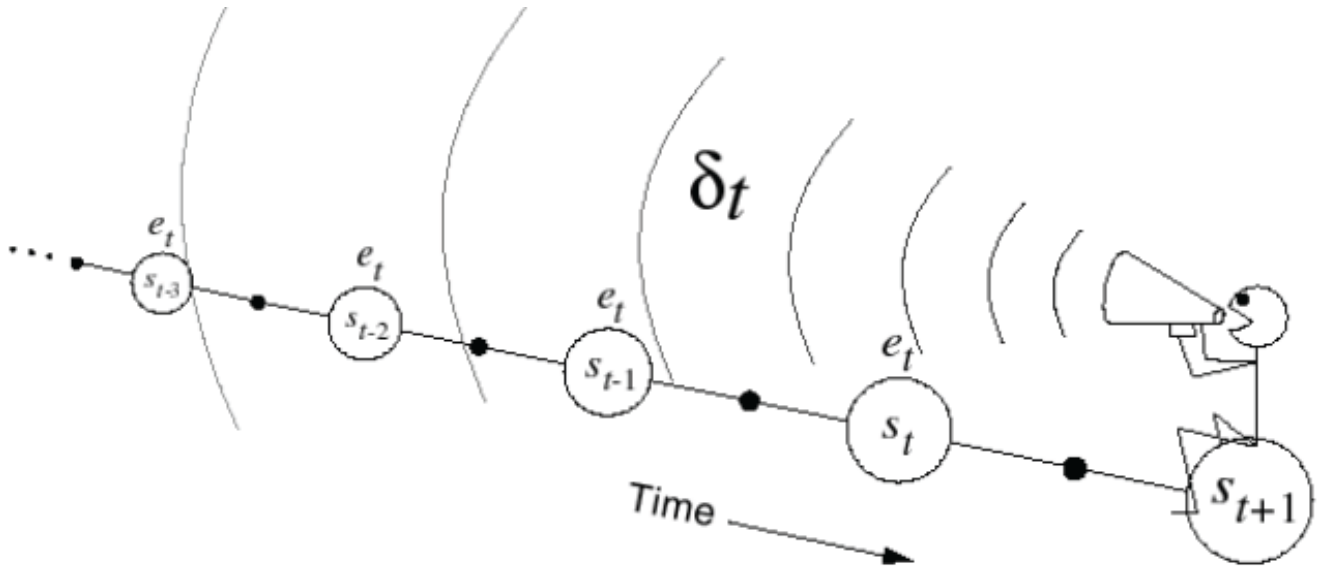
如何在资格迹中观察到当前时刻之前，各个状态的位置?

t=4:	$S_1, S_2, S_2, S_1,$	$E_4(S_1)=r\lambda E_3(S_1)+1=(r\lambda)^3+1;$ $=(r\lambda)^3+(r\lambda)^0$	$E_4(S_2)=r\lambda E_3(S_2) =(r\lambda)^2+(r\lambda)$ $=(r\lambda)^2+(r\lambda)^1;$

;

时序差分值估计

TD(λ)



时序差分值估计

TD(λ)

λ∈[0，1]：　TD(0), TD(λ), TD(1)

- **TD(0)**

$$E_t(s) = \mathbf{1}(S_t = s)$$

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

This is exactly equivalent to TD(0) update

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

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

t=0:		$E_0(S_1)=0$;	$E_0(S_2)=0$
t=1:	S_1	$E_1(S_1)= 1$	$E_1(S_2) =0$
t=2:	S_1, S_2,	$E_2(S_1)=0$	$E_2(S_2)= 1$
t=3:	S_1, S_2, S_2,	$E_3(S_1)=0$	$E_3(S_2)= 1$
t=4:	S_1, S_2, S_2, S_1,	$E_4(S_1)= 1$	$E_4(S_2)=0$
t=5:	S_1, S_2, S_2, S_1, S_1	$E_5(S_1)= 1$	$E_5(S_2)=0$

时序差分值估计

TD(λ)

TD(λ)

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

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

TD(1)

TD(1) eligibility trace discounts time since visit,

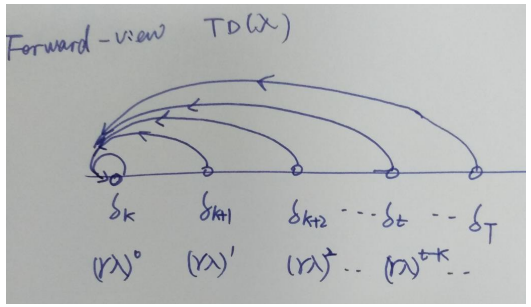
$$E_t(s) = \gamma E_{t-1}(s) + \mathbf{1}(S_t = s)$$

When $\lambda = 1$, sum of TD errors telescopes into MC error

时序差分值估计

TD(λ)

证明：TD(1) 等价于 MC



$$\alpha\left(G_k^\lambda - V(S_k)\right) = \alpha\sum_{t=k}^{T}(\gamma\lambda)^{t-k}\delta_t \quad = \alpha\sum_{t=k}^{T-1}\gamma^{t-k}\delta_t \quad (\delta_T = 0)$$

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

λ=1

$$\begin{aligned}
&\delta_k + \gamma\delta_{k+1} + \gamma^2\delta_{k+2} + \ldots + \gamma^{T-1-k}\delta_{T-1} \\
&= R_{k+1} + \gamma V(S_{k+1}) - V(S_k) \\
&+ \gamma R_{k+2} + \gamma^2 V(S_{k+2}) - \gamma V(S_{k+1}) \\
&+ \gamma^2 R_{k+3} + \gamma^3 V(S_{k+3}) - \gamma^2 V(S_{k+2}) \\
&\quad\vdots \\
&+ \gamma^{T-1-k} R_T + \gamma^{T-k} V(S_T) - \gamma^{T-1-k} V(S_{T-1}) \\
&= R_{k+1} + \gamma R_{k+2} + \gamma^2 R_{k+3} \ldots + \gamma^{T-1-k} R_T - V(S_k) \\
&= G_k - V(S_k)
\end{aligned}$$

第三章 值估计

3.1 随机逼近

3.2 蒙特卡洛值估计

3.3 时序差分值估计

3.4 算法总结

算法总结

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

本讲参考文献

1. Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction. (Second edition, in progress，draft.
2. David Silver，Slides@《Reinforcement Learning:An Introduction》,2016.
3. Simon Haykin，申富饶等译，神经网络与学习机器，第三版。