535514: Reinforcement Learning
Lecture 10 — Model-Free Prediction

Ping-Chun Hsieh

March 24, 2024

3 Major Approaches for Model-Free Prediction

1. Monte Carlo (MC)

2. Temporal Difference: TD(0) and n-step TD

3. TD(λ) and GAE

References:

Richard Sutton and Andrew Barto, Reinforcement Learning: An Introduction, 2019

Singh and Sutton, “Reinforcement Learning with Replacing Eligibility Traces,” ML 1996

Schulman et al., High-Dimensional Continuous Control Using Generalized Advantage Estimation, ICLR 2016

Monte-Carlo for Policy Evaluation

- Recall: Monte-Carlo policy gradient
  - Use sample return G_t in the estimate of policy gradient

$$\nabla_\theta V^{\pi_\theta}(\mu) \approx \sum_{t=0}^{\infty} \gamma^t G_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

- Question: Can we use the same idea for policy evaluation (i.e., finding $V^\pi(s)$ or $Q^\pi(s, a)$)?

Monte-Carlo (MC) Method for Policy Evaluation

To find the value function V^π under a fixed policy π:

Monte-Carlo Policy Evaluation

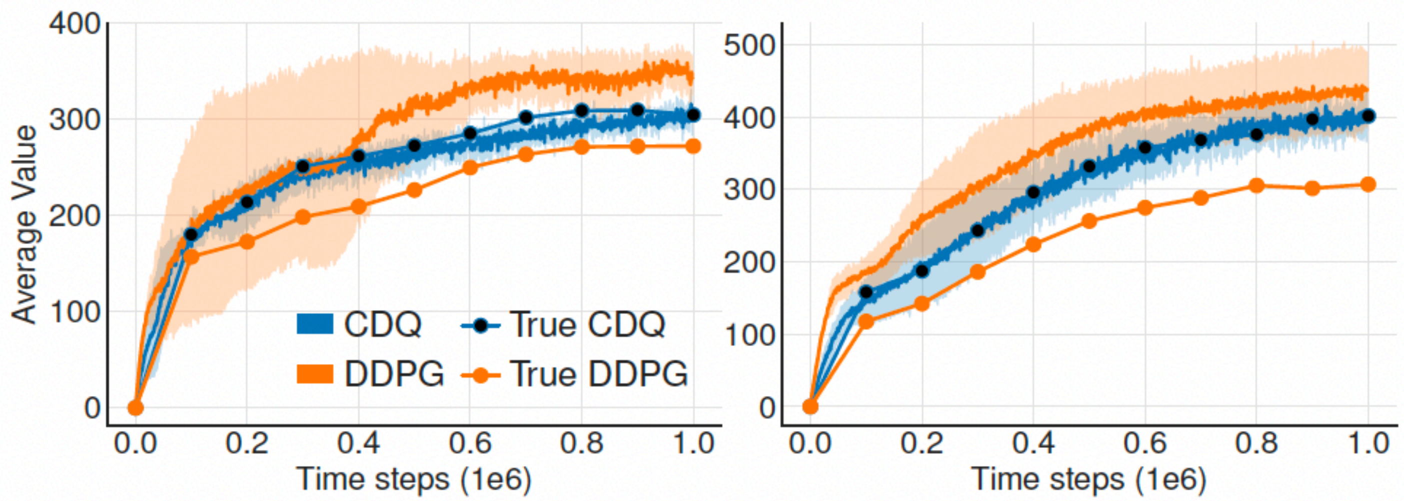
- For episodic environments ➡ sample a set of trajectories $\{\tau^{(i)}\}_{i=1}^{K}$ and calculate average returns $\frac{1}{K}\sum_{i=1}^{K} G(\tau^{(i)}) \approx V^\pi(\mu)$

- For continuing environments ➡ sample a set of trajectories (but with proper ***truncation***) and calculate average returns as an estimate of $V^\pi(\mu)$

Is MC Policy Evaluation Useful in Practice?

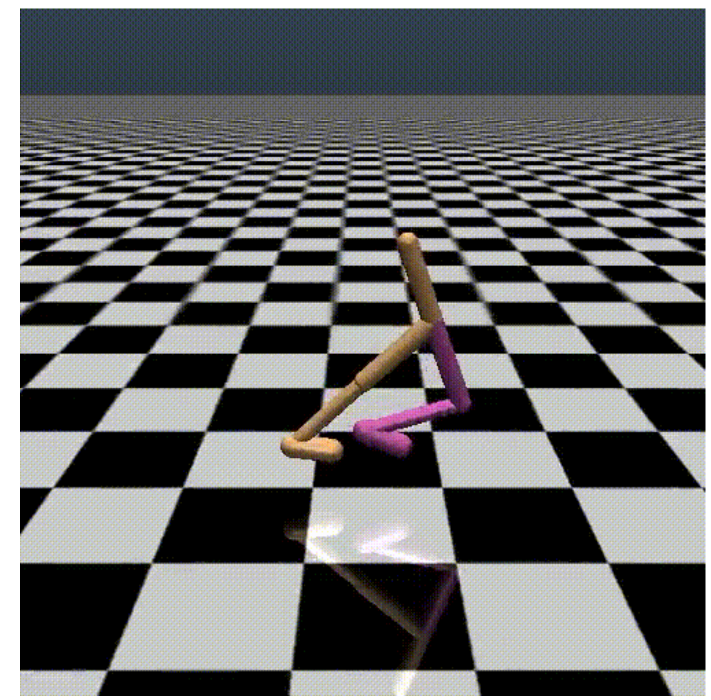
Yes! MC serves as a pseudo-oracle for true $V^{\pi}(s)$ or $Q^{\pi}(s, a)$

Example: Finding the "true value functions" in the TD3 paper



(a) Hopper-v1 (b) Walker2d-v1

Figure 1. Measuring overestimation bias in the value estimates of DDPG and our proposed method, Clipped Double Q-learning (CDQ), on MuJoCo environments over 1 million time steps.



- If the policy is *deterministic*, how many trajectories do we need?
- What if the policy is *stochastic*?

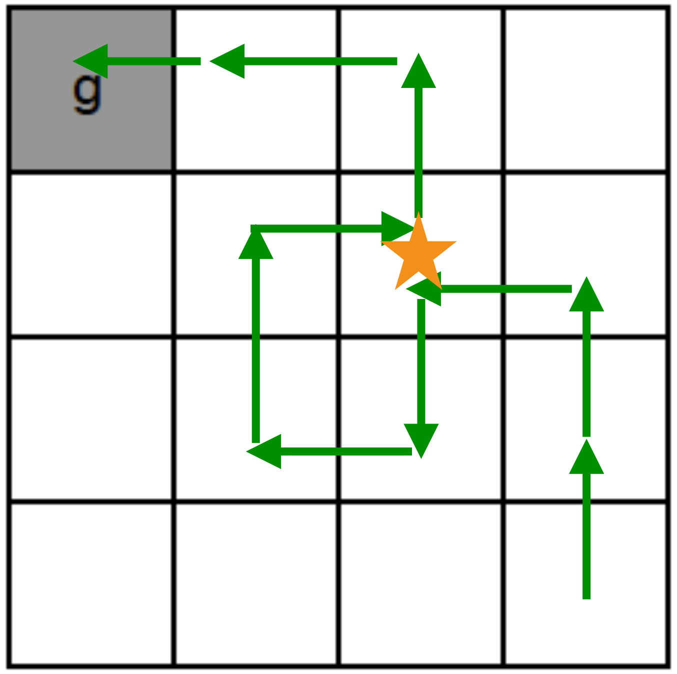
Fujimoto et al., Addressing Function Approximation Error in Actor-Critic Methods, ICML 2018

Two Variants of MC Policy Evaluation: First-Visit and Every-Visit

- A visit to s: an occurrence of a state s in an episode

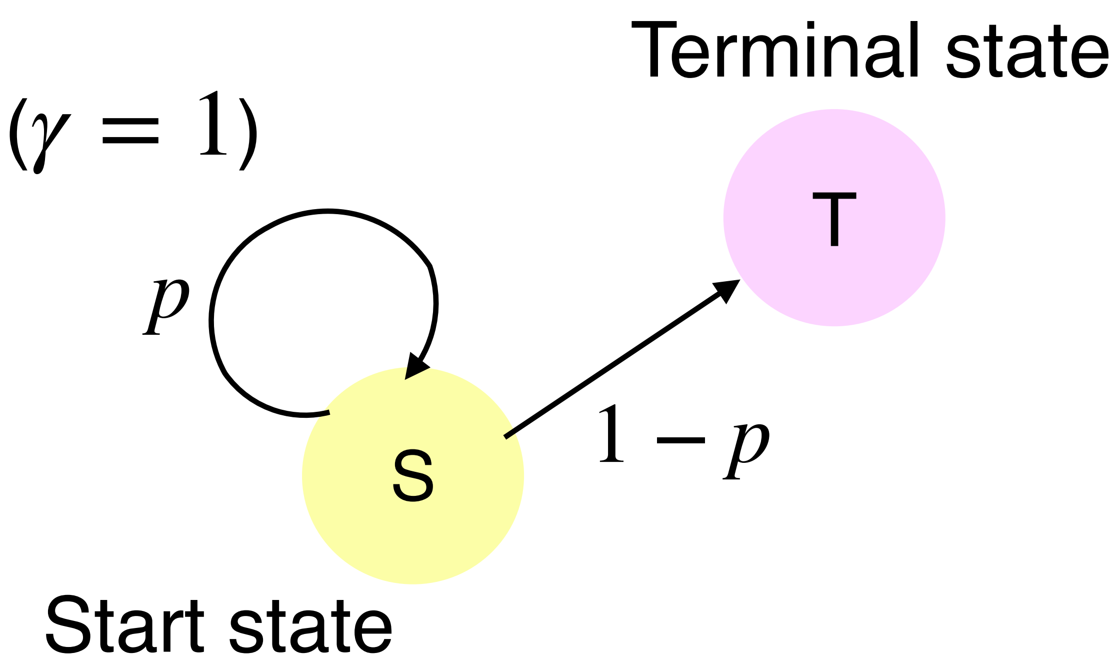
- First-visit MC: Estimate the value of a state as the average of the returns that have followed the first visit to the state in an episode

- Every-visit MC: Estimate the value of a state as the average of the returns that have followed all visits to the state



Example: First visit to ★? How many visits to ★?

Example: 2-State MRP



@Start state: reward = 1

@Terminal state: reward = 0

- Consider a sample trajectory: $S \to S \to S \to S \to T$
- Question: First-visit MC estimate of $V(S)$ = ? 4
- Question: Every-visit MC estimate of $V(S)$ = ?

$$(4 + 3 + 2 + 1)/4 = 2.5$$

- Question: Which estimate is better?

First-Visit MC Policy Evaluation (Formally)

Initialize $N(s) = 0,\ G(s) = 0\ \forall s \in S$
Loop

- Sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$
- Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i - 1} r_{i,T_i}$ as return from time step t onwards in ith episode
- For each state s visited in episode i
  - For **first** time t that state s is visited in episode i
    - Increment counter of total first visits: $N(s) = N(s) + 1$
    - Increment total return $G(s) = G(s) + G_{i,t}$
    - Update estimate $V^\pi(s) = G(s)/N(s)$

(Slide Credit: Emma Brunskill)

Every-Visit MC Policy Evaluation (Formally)

Initialize $N(s) = 0,\ G(s) = 0\ \forall s \in S$
Loop

- Sample episode $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \ldots, s_{i,T_i}$
- Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \cdots \gamma^{T_i-1} r_{i,T_i}$ as return from time step t onwards in ith episode
- For each state s visited in episode i
  - For **every** time t that state s is visited in episode i
    - Increment counter of total first visits: $N(s) = N(s) + 1$
    - Increment total return $G(s) = G(s) + G_{i,t}$
    - Update estimate $V^\pi(s) = G(s)/N(s)$

(Slide Credit: Emma Brunskill)

An Incremental Expression of Sample Mean

- Let $z_1, z_2, z_3 \cdots$ be a sequence of real numbers
- Sample mean of $z_1, \cdots, z_n$ is denoted by $\bar{z}_n$

$$
\begin{aligned}
\bar{z}_n := \frac{1}{n}\sum_{k=1}^{n} z_k &= \frac{1}{n}\left(z_n + \sum_{k=1}^{n-1} z_k\right) \\
&= \frac{1}{n}\left(z_n + (n-1)\bar{z}_{n-1}\right) \\
&= \frac{1}{n}\left(z_n + (n-1)\bar{z}_{n-1} + \bar{z}_{n-1} - \bar{z}_{n-1}\right) \\
&= \bar{z}_{n-1} + \frac{1}{n}\left(z_n - \bar{z}_{n-1}\right)
\end{aligned}
$$

Incremental Monte-Carlo Updates

(Alternative expression of every-visit MC)

- Update $V^{\pi}(s)$ incrementally after each episode

  $s_0, a_0, r_1, \cdots, s_{T-1}, a_{T-1}, r_T$

  For each state s_t with sample 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 environments, we may instead track the exponential moving average (i.e. forget old episodes) by

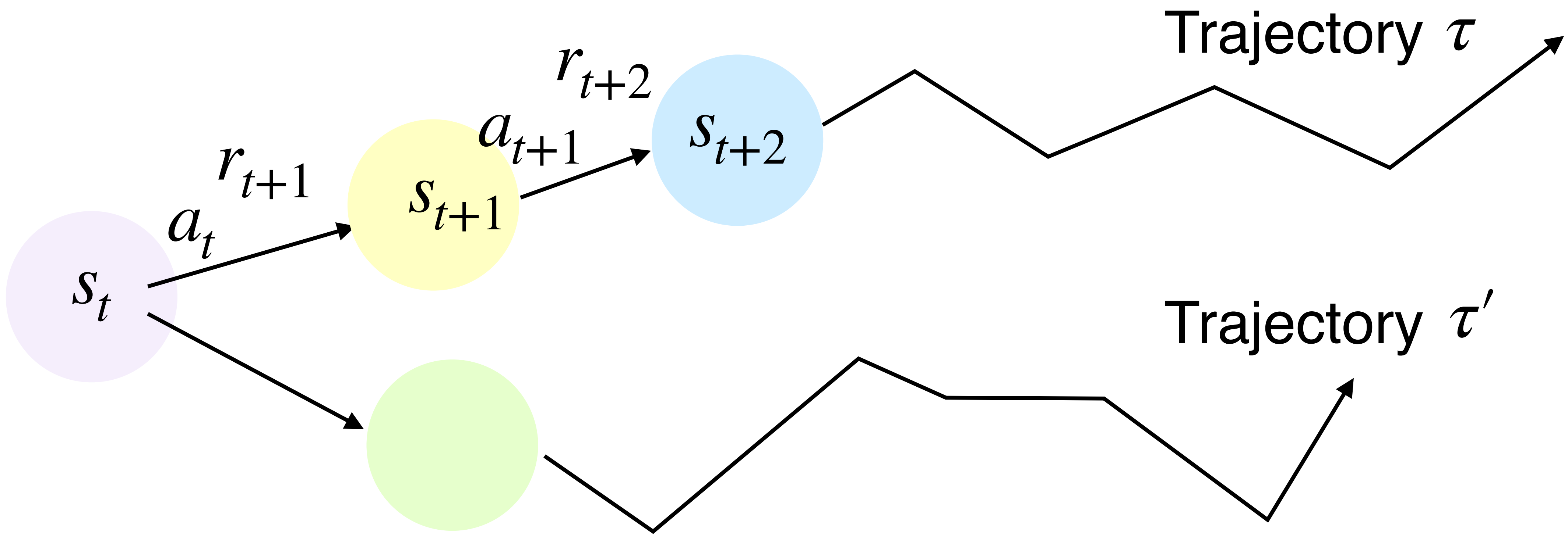
$$V(s_t) \leftarrow V(s_t) + \alpha(G_t - V(s_t))$$

Comparison of First-Visit and Every-Visit MC

1. First-visit MC provides an unbiased estimate

2. Every-visit MC provides a biased but consistent estimate

Why is First-Visit MC Unbiased?



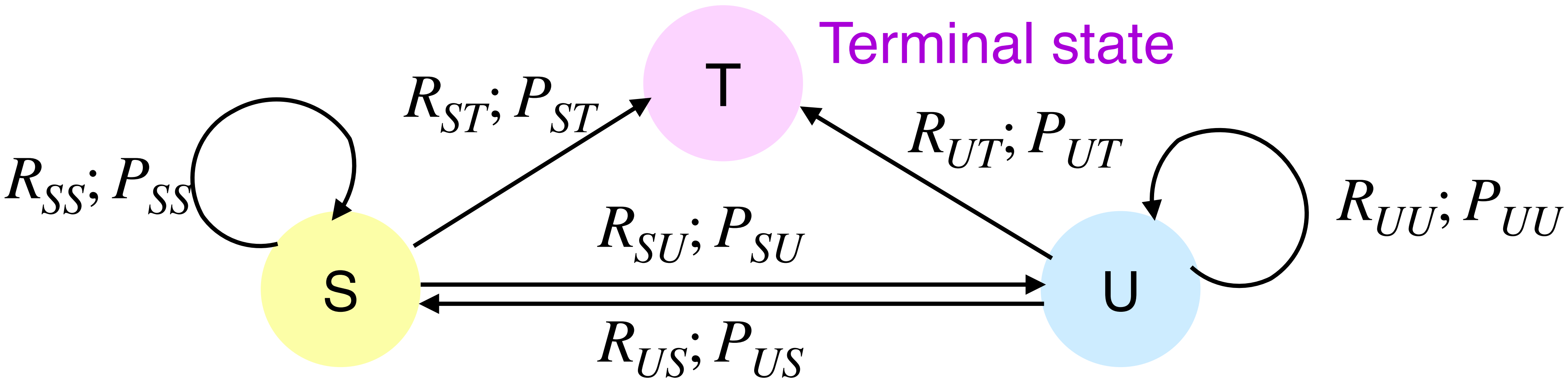
(For simplicity, suppose we use 1 trajectory τ for first-visit MC)

- In trajectory τ, suppose the first visit to state s occurs at time t
- Sample return $G_t(\tau) = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} \cdots$
- Construct a first-visit MC estimate of $V^\pi(s)$ by $G_t(\tau)$

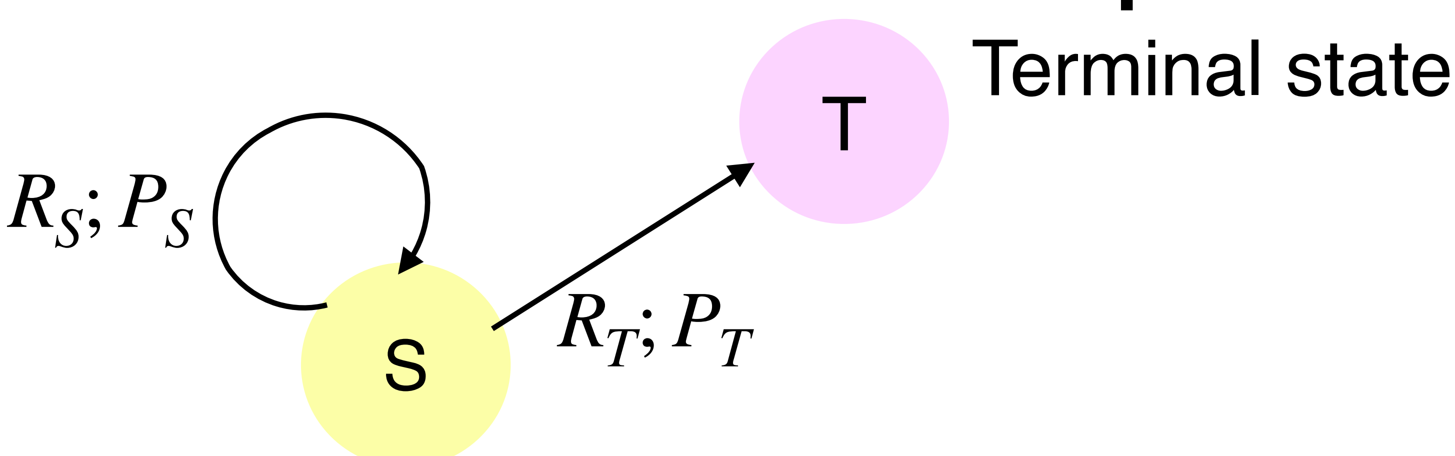
- Question: Do we have $E[G_t(\tau) | s_t = s; \pi] = V^\pi(s)$?
- Question: Does this hold if we use multiple trajectories for first-visit MC?

How to Analyze Every-Visit MC? A Reduction Trick

- Example: Estimating $V(S)$ of a 3-state MRP (assume $\gamma = 1$)



- Idea: Reduce MRP to an **equivalent 2-state MRP** (in what sense?)



$P_S =$

$P_T =$

P_S = prob. of visiting S again before reaching T

P_T = prob. of visiting T before visiting S again

$R_S =$

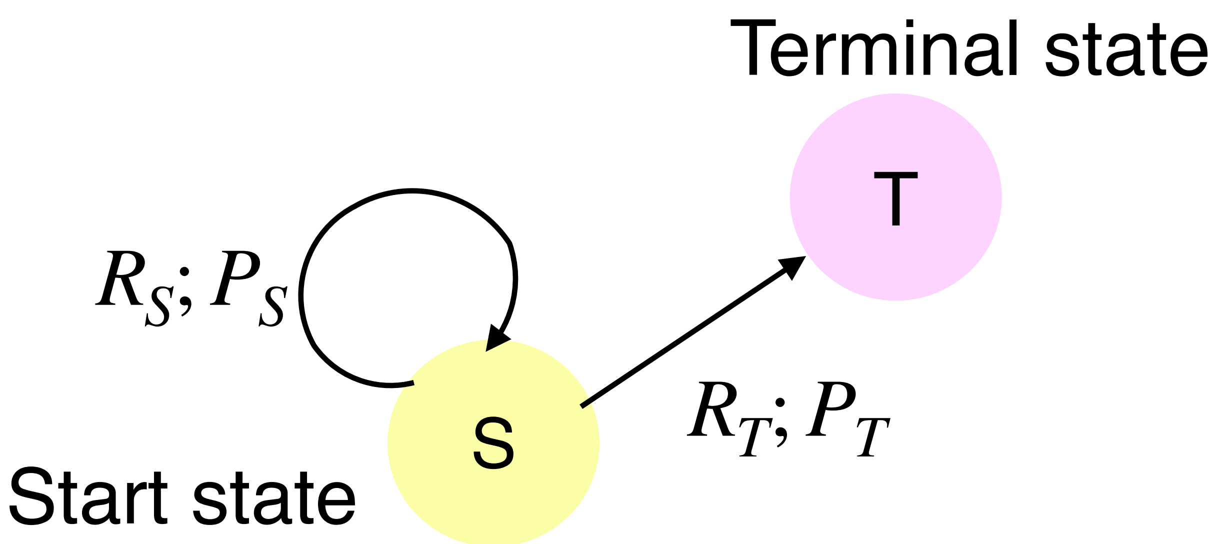
R_S = expected reward of $S \rightsquigarrow S$ transition

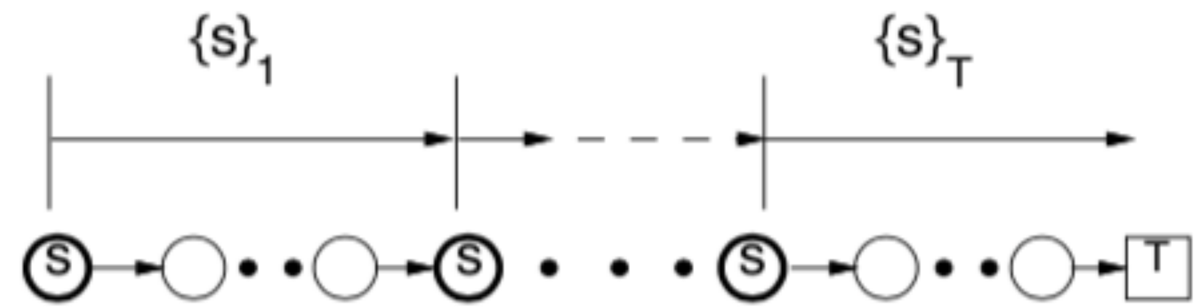
R_T = expected reward of $S \rightsquigarrow T$ transition

$R_T =$

Why is Every-Visit MC is *Biased*?

- Next: Estimate $V(S)$ of **any** MRP (assume $\gamma = 1$)
- Idea: Reduce the MRP to an equivalent 2-state MRP





(A trajectory of the original MRP)

Fact: True value function: $V(S) = \dfrac{P_S}{1 - P_S} R_S + R_T = \dfrac{P_S}{P_T} R_S + R_T$

- Question: Expected every-visit MC estimate over 1 trajectory = ?

$$\sum_{k=0}^{\infty} P_T P_S^k \left(\frac{R_S + 2R_S + \cdots + kR_S + (k+1)R_T}{k+1} \right) = \frac{P_S}{2P_T} R_S + R_T$$

Singh and Sutton, “Reinforcement Learning with Replacing Eligibility Traces,” ML 1996

Why is Every-Visit MC is *Biased*? (Cont.)

- We use the same notations as in the previous page

- **Every-Visit MC is Biased but Consistent in the Limit**:

The expected every-visit MC estimate after n episodes is

$$\frac{nP_S}{(n+1)P_T}R_S + R_T$$

Thus, every-visit MC estimate is <u>biased</u> and the amount of bias is

$$\frac{P_S}{(n+1)P_T}R_S$$

Moreover, every-visit MC is consistent in the limit $n \to \infty$

Singh and Sutton, "Reinforcement Learning with Replacing Eligibility Traces," ML 1996

Any Issue With Monte-Carlo Policy Evaluation?

1. MC is applicable mainly to episodic problems

- For continuing problems, truncation of trajectories is needed (but may incur some bias)

2. MC can only learn from complete sequences

3. MC generally has high variance

- requires a lot of samples for convergence
- might be impractical in the low-data regime

Temporal Difference (TD)

Motivating Example: A Commuter's Daily Life

- A commuter travels from NYCU back to Taichung after work

State	Old Estimate of Time-to-Go	Sample Elapsed Time for "Today"	Predicted Total Time as of Now
Leaving office	65	0	
Taking the bus	55	8	
Reaching THSR Hsinchu Station	40	33	
Reaching THSR Taichung Station	10	64	
Arriving home	0	76	

- What technique are we using here? *Bootstrapping!*

What is TD? Comparison: TD(0) vs MC

- Goal: Evaluate V^{π} under a fixed policy π

1. Incremental every-visit Monte-Carlo: use sample return

$$V_{k+1}(s_t) \leftarrow V_k(s_t) + \alpha(G_t - V_k(s_t))$$

2. Temporal difference algorithm TD(0): use estimated return

$$V_{k+1}(s_t) \leftarrow V_k(s_t) + \alpha\left(r_{t+1} + \gamma V_k(s_{t+1}) - V_k(s_t)\right)$$

  - $r_{t+1} + \gamma V_k(s_{t+1})$ is the estimated return (called TD target)
  - $r_{t+1} + \gamma V_k(s_{t+1}) - V_k(s_t) =: \delta_t$ is called the TD error

MC Error and TD Error

- Fact: MC error can be written as a sum of TD errors

$$
\begin{aligned}
\underbrace{G_t - V_k(s_t)}_{\text{MC error}} &= \left(r_{t+1} + \gamma G_{t+1}\right) - V_k(s_t) + \gamma V_k(s_{t+1}) - \gamma V_k(s_{t+1}) \\
&= \delta_t + \gamma\left(G_{t+1} - V_k(s_{t+1})\right) \\
&= \delta_t + \gamma \delta_{t+1} + \cdots + \gamma^{T-t-1}\delta_{T-1} + \gamma^{T-t}\left(\underbrace{G_T - V_k(s_T)}_{=0-0}\right) \\
&= \sum_{k=t}^{T-1} \gamma^{k-t}\delta_k
\end{aligned}
$$

Terminal state

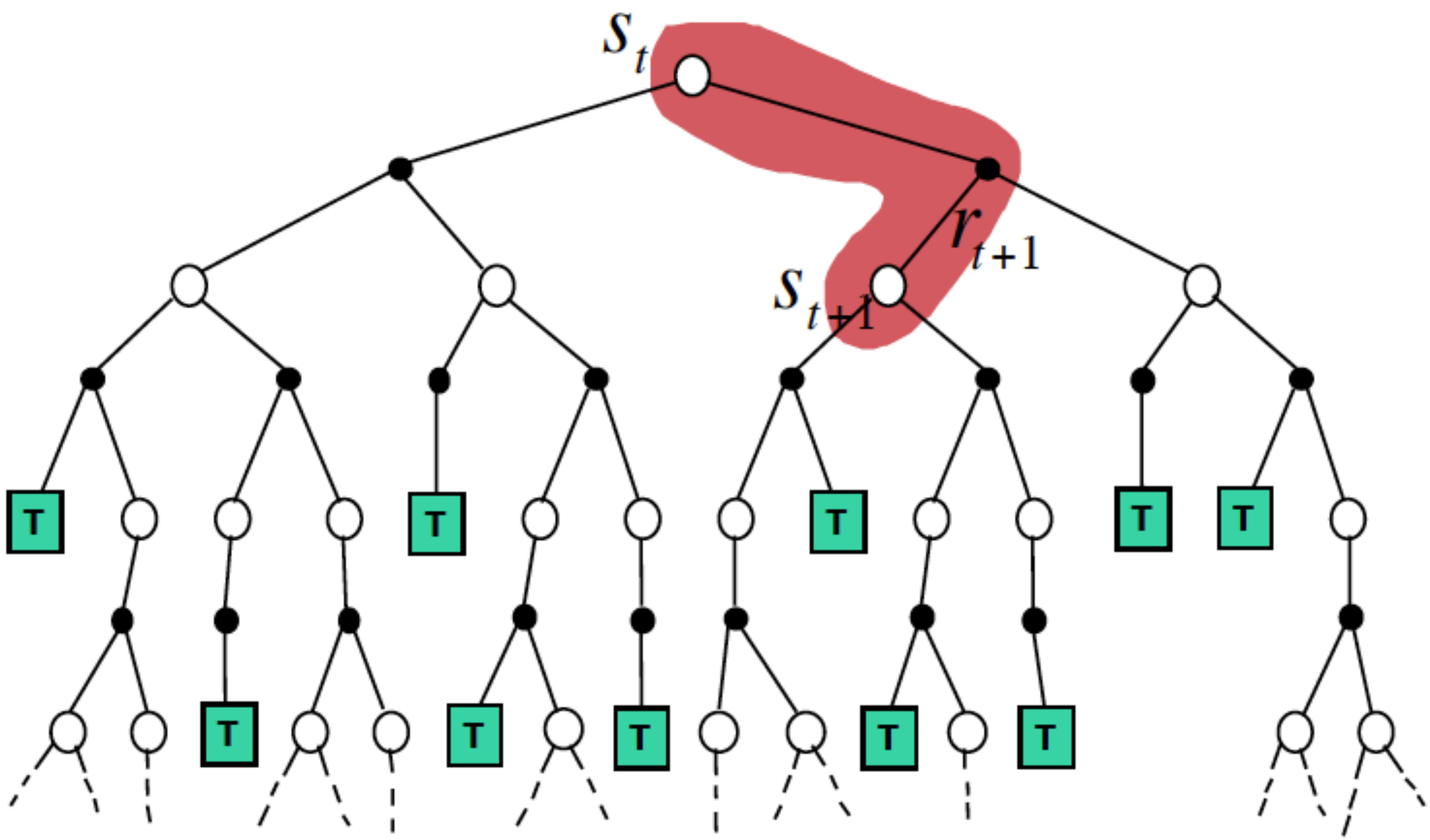
- Question: Why is the above observation intuitively useful?

Features of TD

1. TD is model-free (why?)
   - TD learns directly from episodes **without** estimating MDP transition probabilities or reward function

2. TD learns from incomplete episodes by <u>bootstrapping</u>
   - TD updates a guess towards a guess
   - Question: Why is this a good feature (compared to MC)?

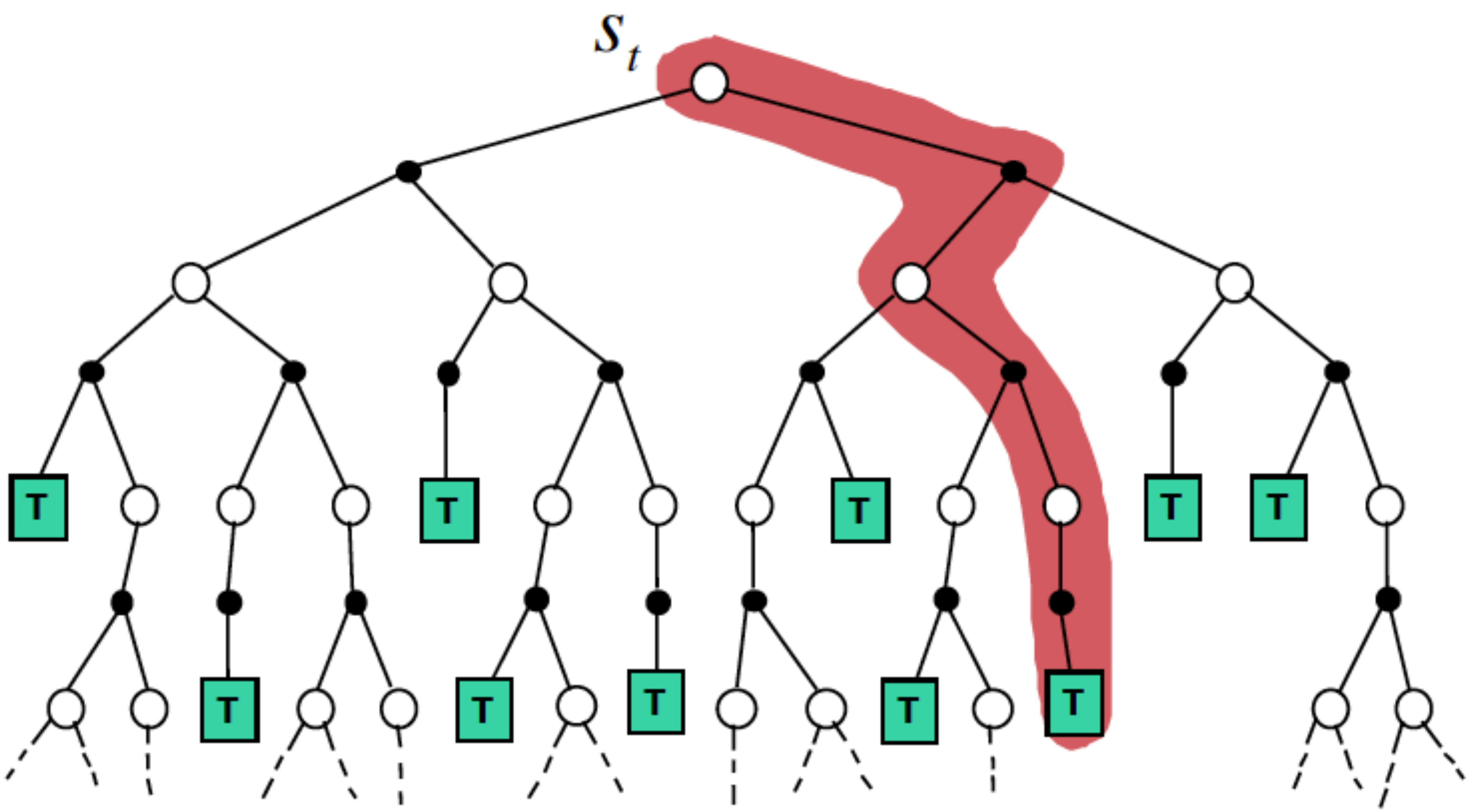
Visualization: TD(0) Backup



$$V_{k+1}(s_t) \leftarrow V_k(s_t) + \alpha\left(r_{t+1} + \gamma V_k(s_{t+1}) - V_k(s_t)\right)$$

(Figure Credit: David Silver)

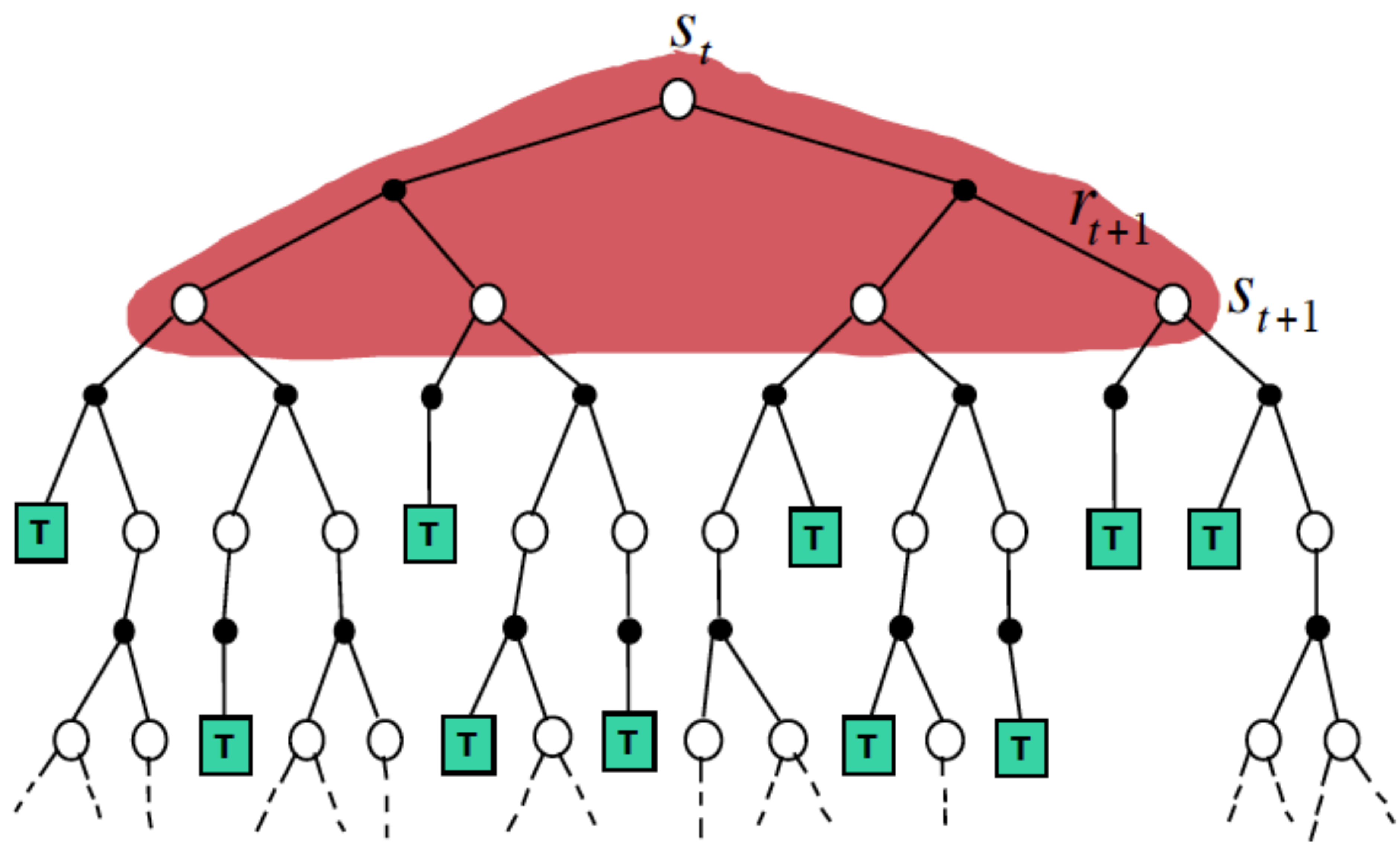
Visualization: Monte-Carlo Backup



$$V_{k+1}(s_t) \leftarrow V_k(s_t) + \alpha(G_t - V_k(s_t))$$

(Figure Credit: David Silver)

Visualization: IPE Operator Backup

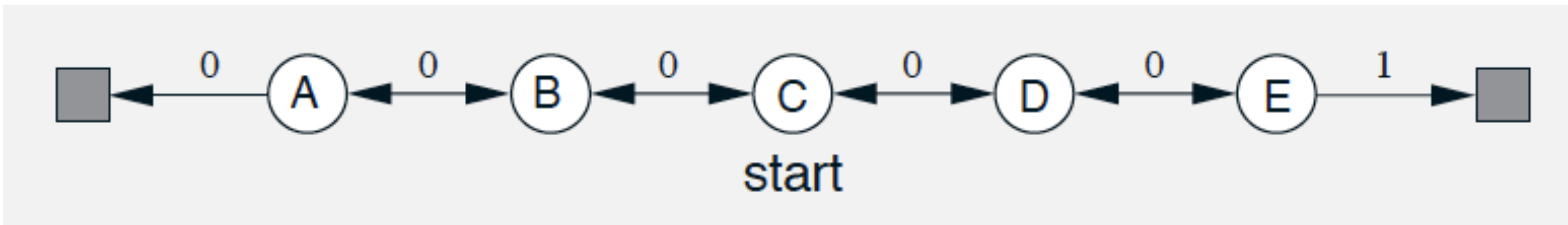


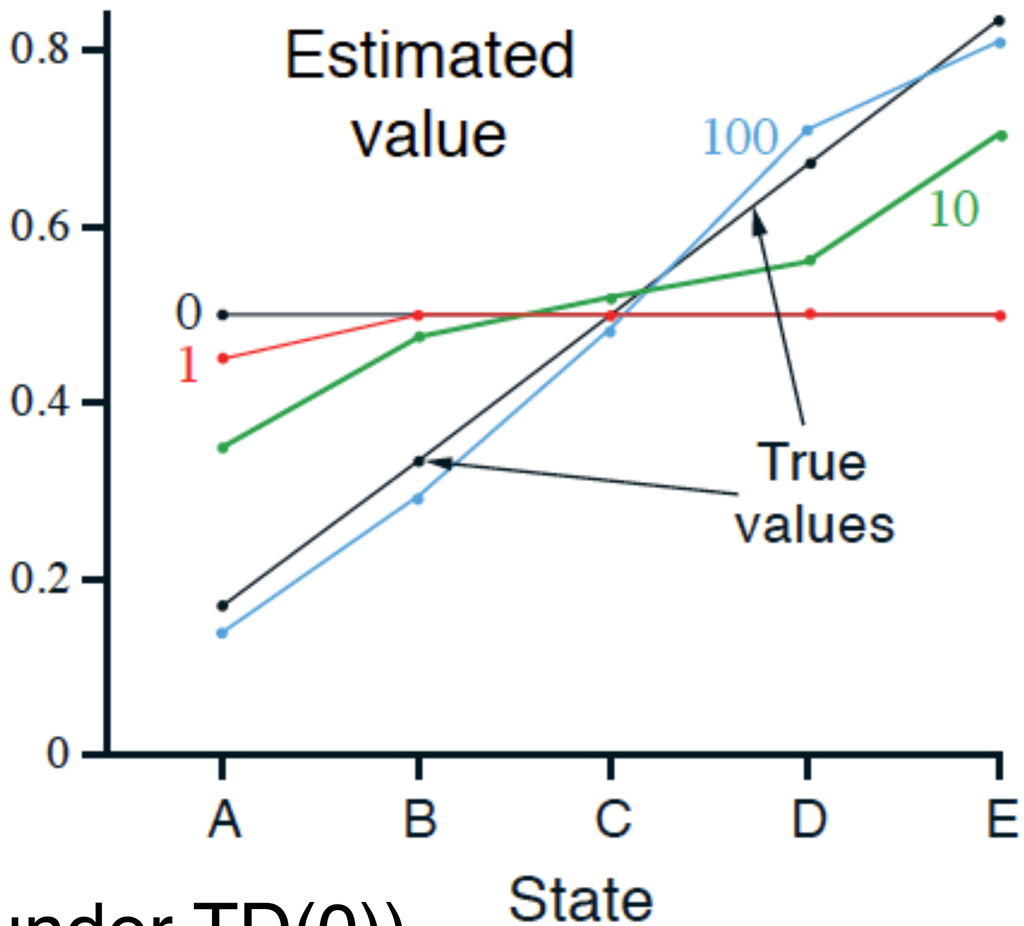
$$V_{k+1}(s_t) \leftarrow \mathbb{E}_{P_\pi}\left[r_{t+1} + \gamma V_k(s_{t+1})\right]$$

(Figure Credit: David Silver)

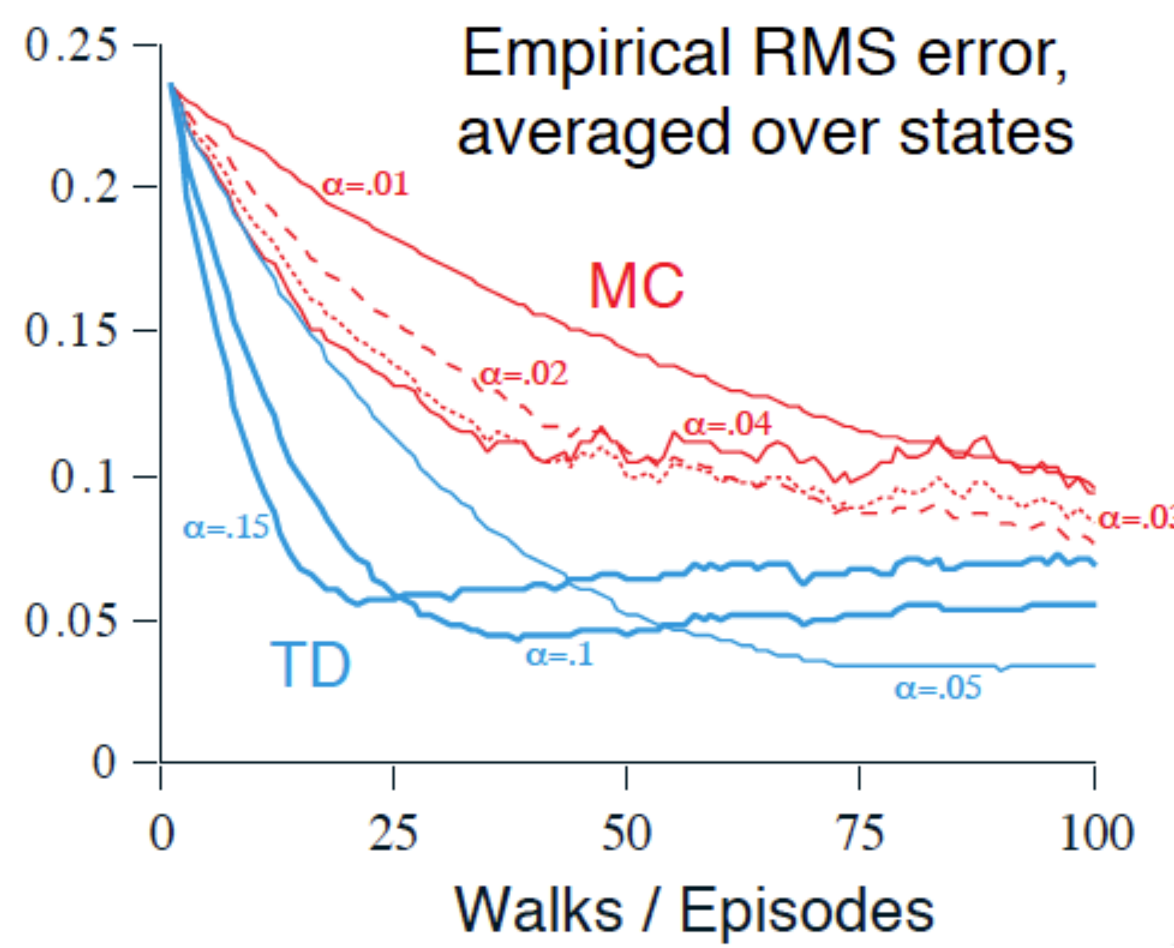
Efficiency: TD(0) vs MC

- Example: Random walk MRP with 5 states





(under TD(0))



(Figure Source: Sutton & Barto, Section 6)

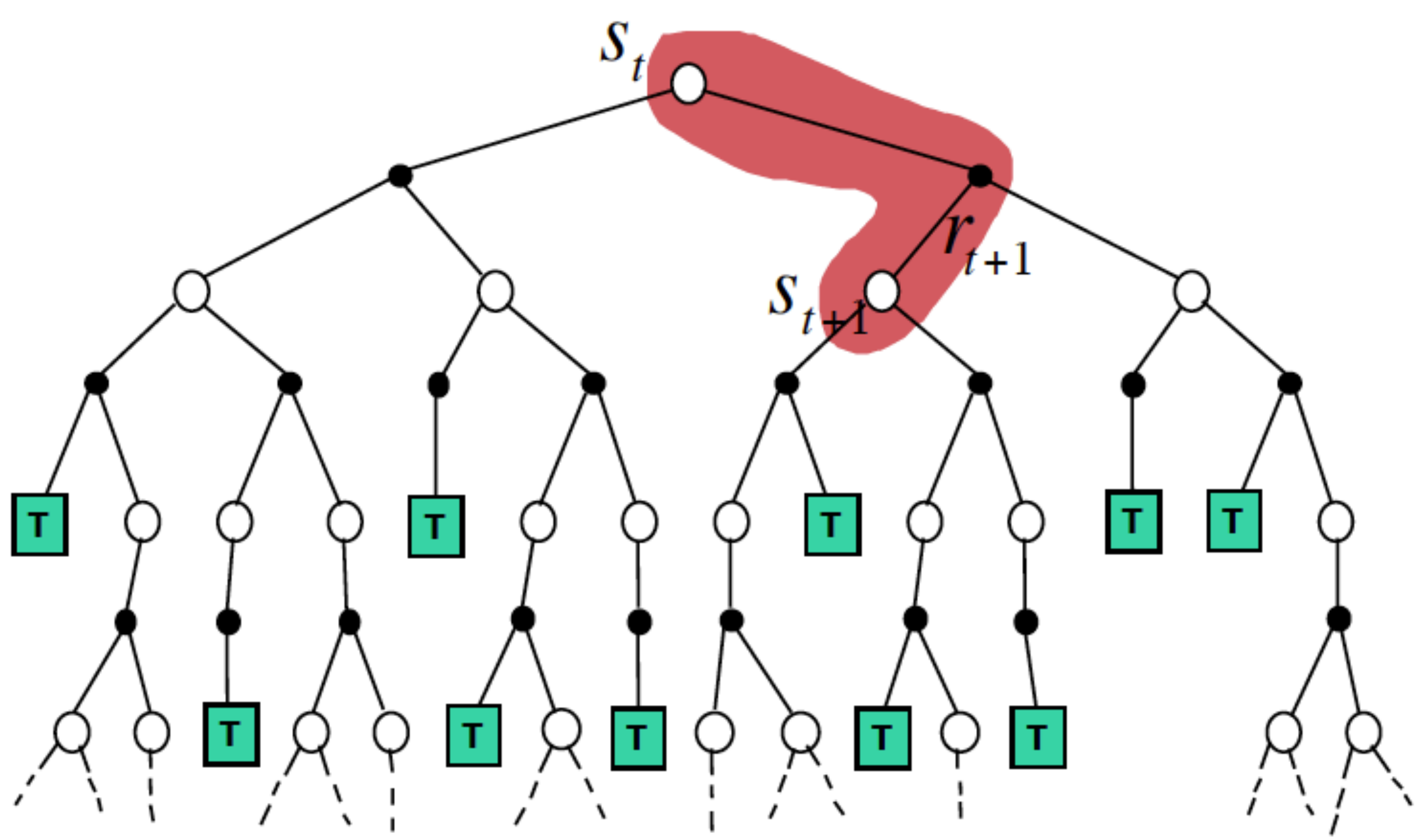
Extension of TD(0): n-Step TD and TD(λ)

Use n-Step Return For Prediction?

- Recall: update rule of TD(0)

$$V_{k+1}(s_t) \leftarrow V_k(s_t) + \alpha\left(r_{t+1} + \gamma V_k(s_{t+1}) - V_k(s_t)\right)$$

- Question: Can we consider n steps into the future?



n-Step Bootstrapping For Prediction (Formally)

- Define the n-step estimated return

$$G_t^{(n)} = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n V(s_{t+n})$$

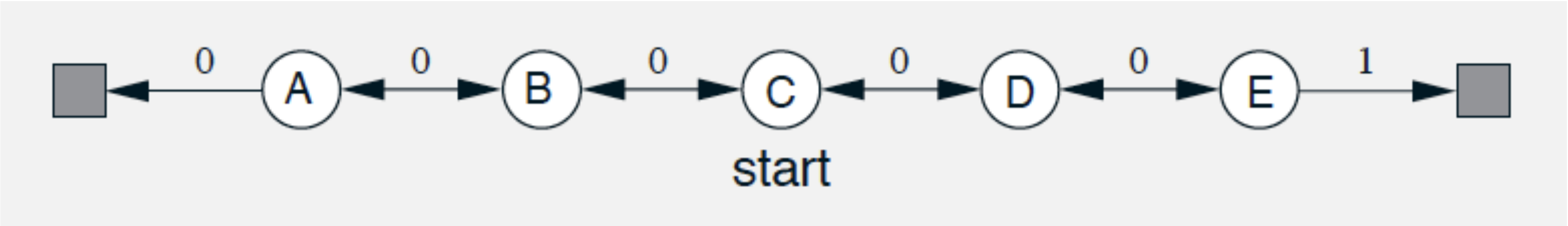
- n-step TD for policy evaluation

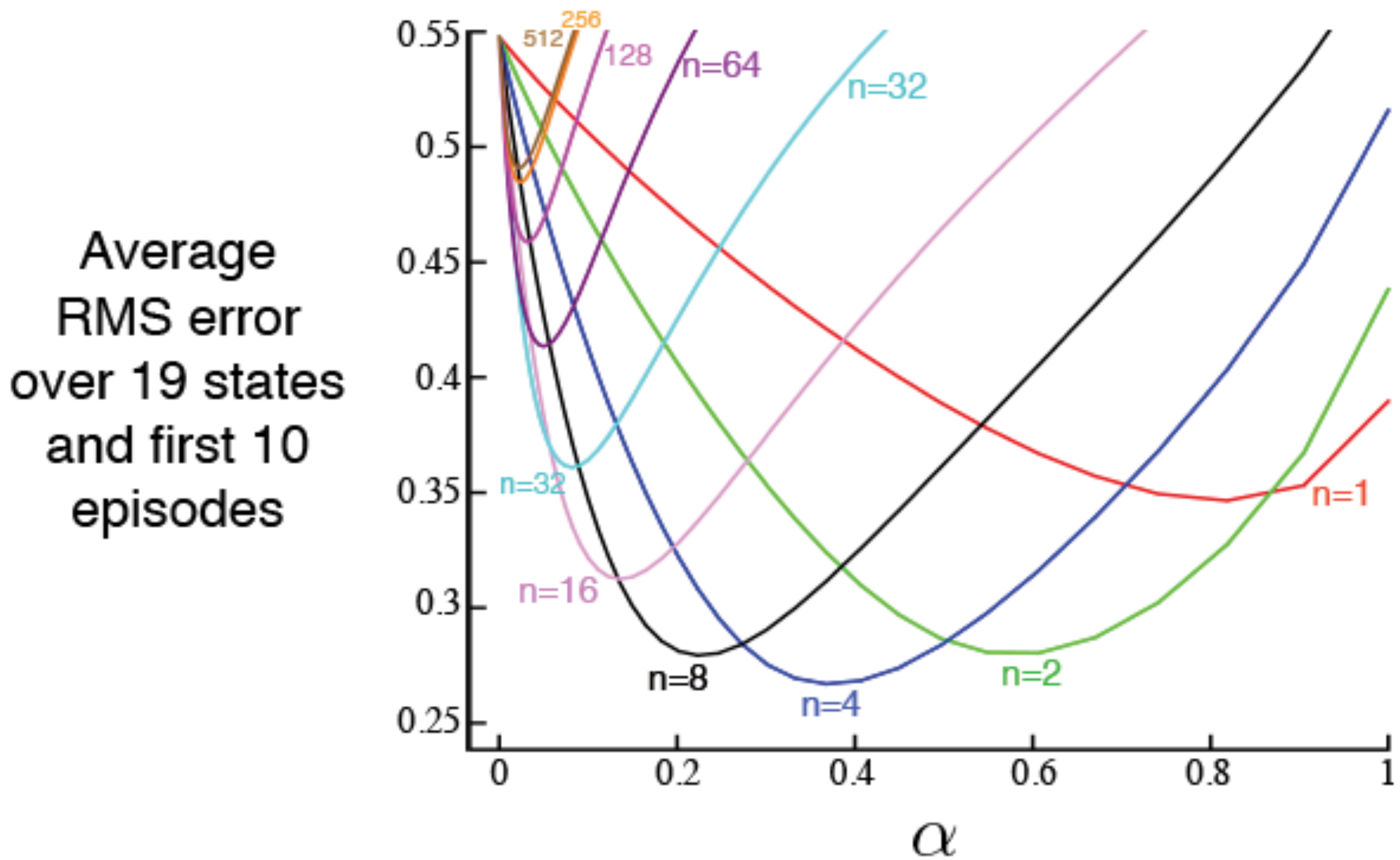
$$V(s_t) \leftarrow V(s_t) + \alpha\left(G_t^{(n)} - V(s_t)\right)$$

- Special case:
  - $n = 1$: standard TD(0)
  - $n = \infty$: MC

Which Value of n is Better?

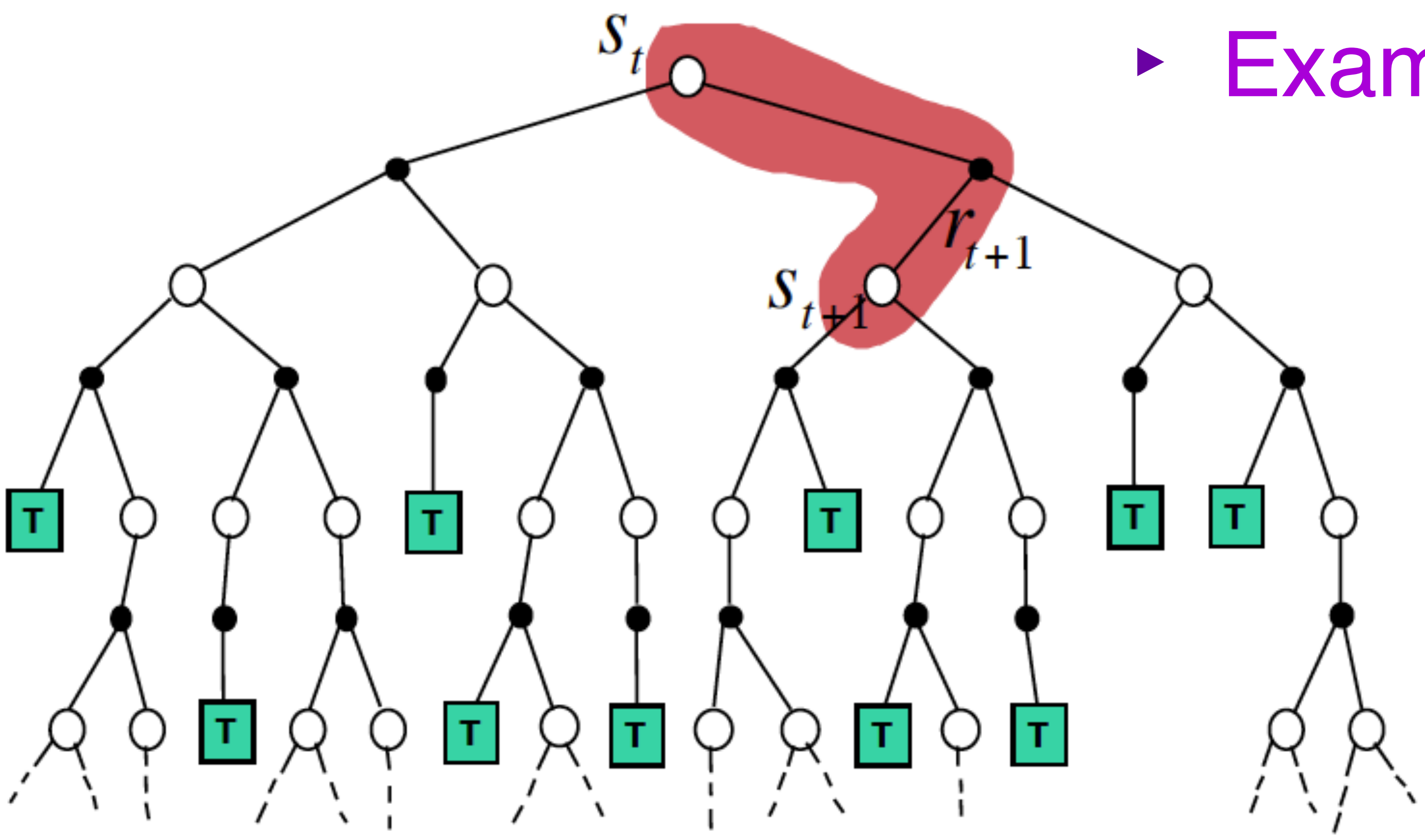
- Example: Random walk MRP with 19 states





(Figure Source: Sutton & Barto, Section 7.1)

Combine n-Step Returns Over Different n?



- Example:

$$G_t^{(1)} = r_{t+1} + \gamma V(s_{t+1})$$

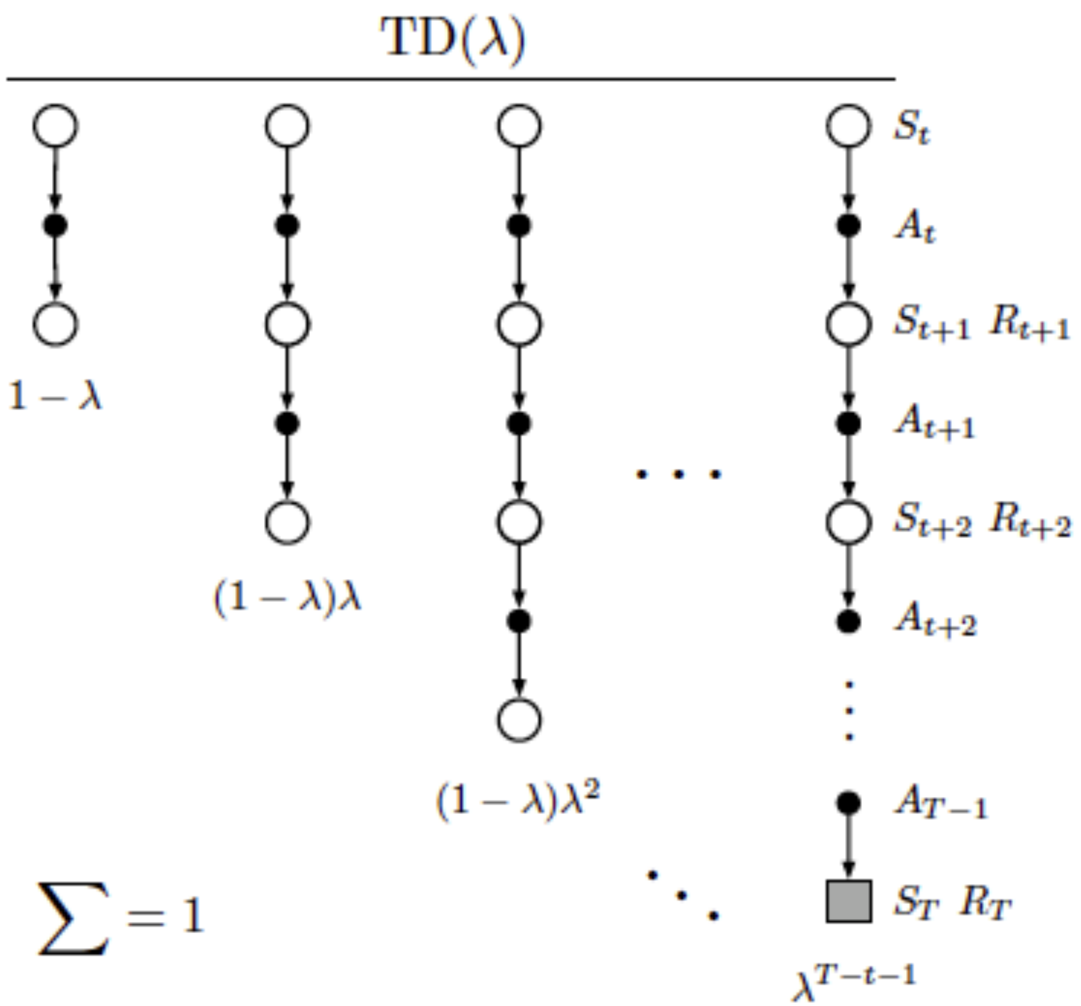
$$G_t^{(2)} = r_{t+1} + \gamma r_{t+2} + \gamma^2 V(s_{t+2})$$

$$\hat{G}_t = \frac{1}{2}\left(G_t^{(1)} + G_t^{(2)}\right)$$

$$V_{k+1}(s_t) \leftarrow V_k(s_t) + \alpha\left(\hat{G}_t - V_k(s_t)\right)$$

- Question: Any systematic way to combine n-Step Returns?

λ-Return and TD(λ)



- Define λ-return as

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

- G_t^{λ} combines all n-step returns using weights $(1-\lambda)\lambda^{n-1}$

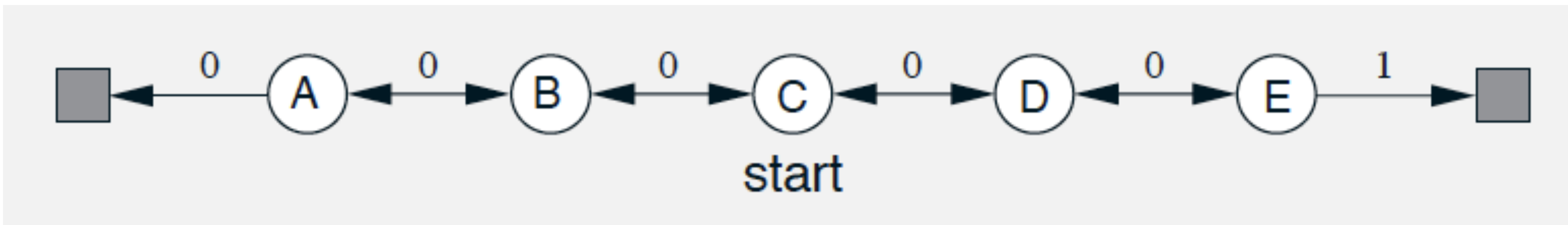
- The update rule of TD(λ) algorithm:

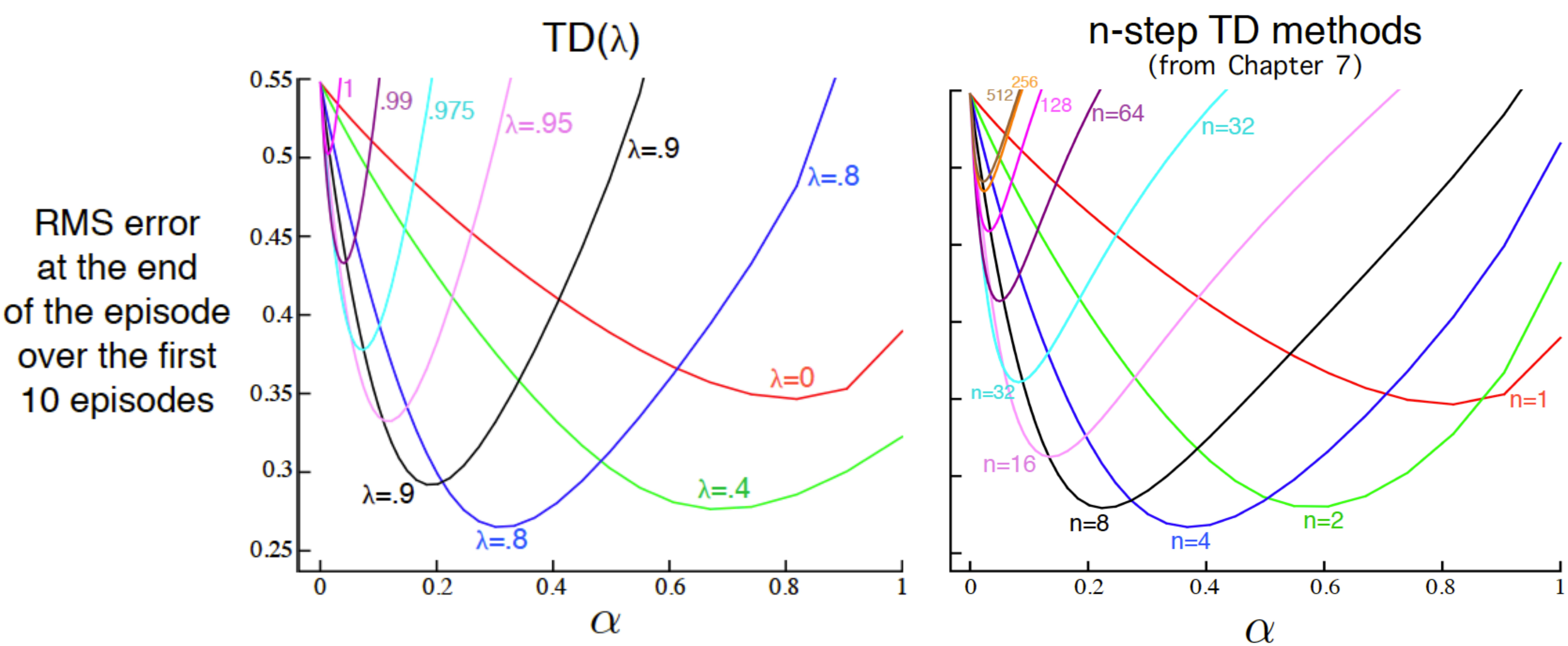
$$V_{k+1}(s_t) \leftarrow V_k(s_t) + \alpha\left(G_t^{\lambda} - V_k(s_t)\right)$$

Richard Sutton, "Learning to predict by the methods of temporal difference," ML 1988

Which Value of λ is Better?

- Example: Random walk MRP with 19 states





(Figure Source: Sutton & Barto, Section 12.1)

Next Question: How to Estimate $A^{\pi}(s, a)$?

Generalized Advantage Estimator (GAE): Using TD(λ) to Estimate $A^{\pi}(s,a)$

Let $V(s)$ be the current estimate of true value $V^{\pi}(s)$

$$\hat{A}_t^{(1)} := r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \qquad (= \delta_t)$$

$$\hat{A}_t^{(2)} := r_{t+1} + \gamma r_{t+2} + \gamma^2 V(s_{t+2}) - V(s_t) \qquad (= \delta_t + \gamma\delta_{t+1})$$

$$\hat{A}_t^{(3)} := r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 V(s_{t+3}) - V(s_t) \qquad (= \delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2})$$

$$\hat{A}_t^{(k)} := r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^k V(s_{t+k}) - V(s_t) \qquad \left(= \sum_{\ell=0}^{k-1} \gamma^{\ell}\delta_{t+\ell}\right)$$

- Fact: $\hat{A}_t^{(\infty)} = \sum_{\ell=0}^{\infty} \gamma^{\ell}\delta_{t+\ell} = G_t - V(s_t)$

- GAE Estimator:

$$\hat{A}_t^{GAE(\gamma,\lambda)} = (1-\gamma)\left(\hat{A}_t^{(1)} + \gamma\hat{A}_t^{(2)} + \gamma^2\hat{A}_t^{(3)} + \cdots\right) = \sum_{\ell=0}^{\infty} (\gamma\lambda)^{\ell}\delta_{t+\ell}$$

Schulman et al., High-Dimensional Continuous Control Using Generalized Advantage Estimation, ICLR 2016

Algorithm: REINFORCE With GAE

Recall: (P5) REINFORCE with advantage

$$\nabla_\theta V^{\pi_\theta}(\mu) = \mathbb{E}_{\tau \sim P_\mu^{\pi_\theta}}\left[\sum_{t=0}^{\infty} \gamma^t A^{\pi_\theta}(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right]$$

- REINFORCE with GAE

Step 1: Initialize θ_0 and step size η

Step 2: Sample a trajectory $\tau \sim P_\mu^{\pi_\theta}$ and make the update as

$$\theta_{k+1} = \theta_k + \eta\left(\sum_{t=0}^{\infty} \gamma^t \hat{A}_t^{GAE(\gamma,\lambda)} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right)$$

where $\hat{A}_t^{GAE(\gamma,\lambda)}$ is constructed from $V(s)$ learned by TD

(Repeat Step 2 until termination)

Some Discussions on GAE

1. Do we need to wait until the end of a trajectory to construct GAE?

2. How to efficiently calculate GAE for different t of the same trajectory?

```
def calculate_advantages(rewards, values, discount_factor, trace_decay, normalize = True):

    advantages = []
    advantage = 0
    next_value = 0

    for r, v in zip(reversed(rewards), reversed(values)):
        td_error = r + next_value * discount_factor - v
        advantage = td_error + advantage * discount_factor * trace_decay
        next_value = v
        advantages.insert(0, advantage)

    advantages = torch.tensor(advantages)
```

3. Where does $V(s)$ in GAE come from?