Model Free Prediction

DAVIDE BACCIU – BACCIU@DI.UNIPI.IT

UNIVERSITÀ DI PISA

Introduction

Outline

- ✓ Introduction
- ✓ Monte-Carlo approaches
- ✓ Temporal-Difference (TD) learning
- ✓ TD(λ)

Model-Free Reinforcement Learning

- ✓So far: solve a known MDP (states, transition, rewards, actions)
- ✓Model free
  - ✓No environment model
  - ✓No knowledge of MDP transition/rewards
- ✓Model-free prediction - Estimate the value function of an unknown MDP
- ✓Model-free control - Optimise the value function of an unknown MDP

Monte-Carlo

Monte-Carlo (MC) Reinforcement Learning

- ✓MC methods learn directly from episodes of experience
- ✓MC is model-free: no knowledge of MDP transitions/rewards
- ✓MC learns from complete episodes: no bootstrapping
- ✓MC uses the simplest possible idea: value = mean return across episodes
- ✓Caveat: can only apply MC to episodic MDPs
  - ✓All episodes must terminate

Monte-Carlo Policy Evaluation

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

$$S_1, A_1, R_2, \ldots, R_k \sim \pi$$

- ✓Recall that return is the total discounted reward

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

- ✓Recall that value function is the expected return

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

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

First-Visit Monte-Carlo Policy Evaluation

- ✓ To evaluate state s
- ✓ The first time-step t that state s is visited in an episode
  I. Increment counter $\mathrm{N}(s) \leftarrow \mathrm{N}(s) + 1$
  II. Increment total return $\mathrm{S}(s) \leftarrow \mathrm{S}(s) + G_t$
  III. Value is estimated by mean return $\mathrm{V}(s) = \mathrm{S(s)}/\mathrm{N}(s)$
- ✓ By law of large numbers

$$\mathrm{V}(s) \rightarrow v_\pi(s) \text{ as } N(s) \rightarrow \infty$$

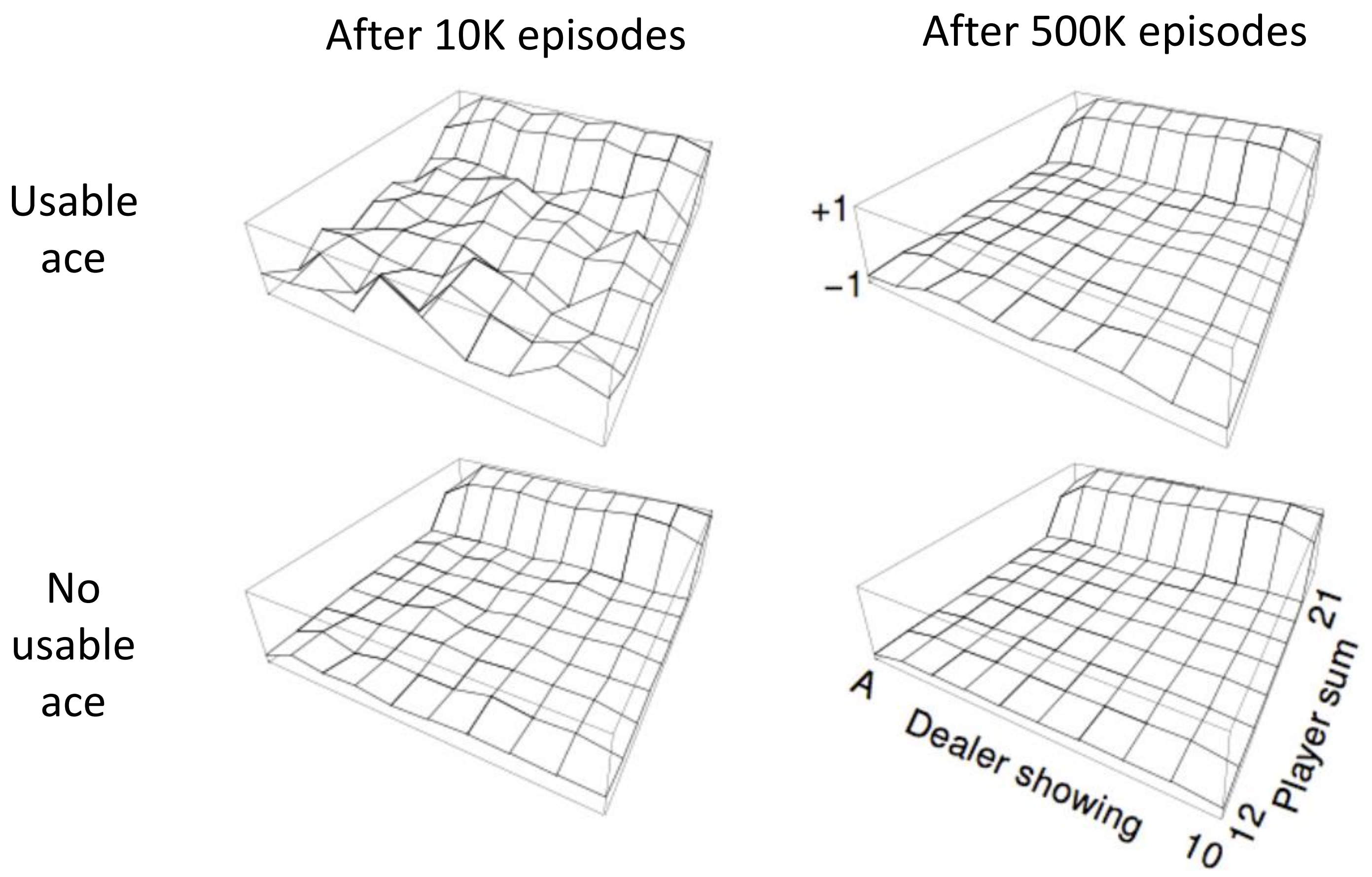
Every-Visit Monte-Carlo Policy Evaluation

- ✓ To evaluate state s
- ✓ Every time-step t that state s is visited in an episode
  I. Increment counter $\mathrm{N}(s) \leftarrow \mathrm{N}(s) + 1$
  II. Increment total return $\mathrm{S}(s) \leftarrow \mathrm{S}(s) + G_t$
  III. Value is estimated by mean return $\mathrm{V}(s) = \mathrm{S}(\mathrm{s})/\mathrm{N}(s)$

Blackjack Example

- ✓States (200 of them):
  - ✓Current sum (12-21)
  - ✓Dealer’s showing card (ace-10)
  - ✓Do I have a useable ace? (yes-no)
- ✓Reward for action stick (Stop receiving cards (and terminate)):
  - ✓+1 if sum of cards > sum of dealer cards
  - ✓0 if sum of cards = sum of dealer cards
  - ✓-1 if sum of cards < sum of dealer cards
- ✓Reward for action twist (Take another card (no replacement)):
  - ✓-1 if sum of cards > 21 (and terminate)
  - ✓0 otherwise
  - ✓Transitions: automatically twist if sum of cards < 12

Blackjack Value Function after MC Learning



Policy: stick if sum of cards ≥ 20, otherwise twist

Incremental Mean

The mean $\mu_1, \mu_2, \ldots$ of a sequence $x_1, x_2, \ldots$ can be computed incrementally

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

$$\mu_k = \frac{1}{k}\left(x_k + (k-1)\mu_{k-1}\right) = \mu_{k-1} + \frac{1}{k}\left(x_k - \mu_{k-1}\right)$$

Incremental Mean MC Update

- ✓ Update $V(s)$ incrementally after episode $S_1, A_1, R_2, \ldots, R_T$
- ✓ For each state S_t with return G_t
  I. Increment counter $N(s) \leftarrow N(s) + 1$
  II. Update value function (with incremental mean)

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

- ✓ In non-stationary problems track a running mean (forget old episodes)

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

Temporal-Difference Learning

Temporal-Difference (TD) Learning

- ✓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

MC Vs TD Learning

- Goal: learn v_π from episodes of experience under policy π
- Incremental every-visit MC
  - Update value $V(S_t)$ toward actual return G_t

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

- Simplest temporal-difference learning algorithm (TD(0))
  - Update value $V(S_t)$ toward estimated return $R_t + \gamma V(S_{t+1})$

$$V(S_t) \leftarrow V(S_t) + \alpha(\underbrace{\underbrace{R_t + \gamma V(S_{t+1})}_{\text{TD target}} - V(S_t)}_{\text{TD error } \delta_t})$$

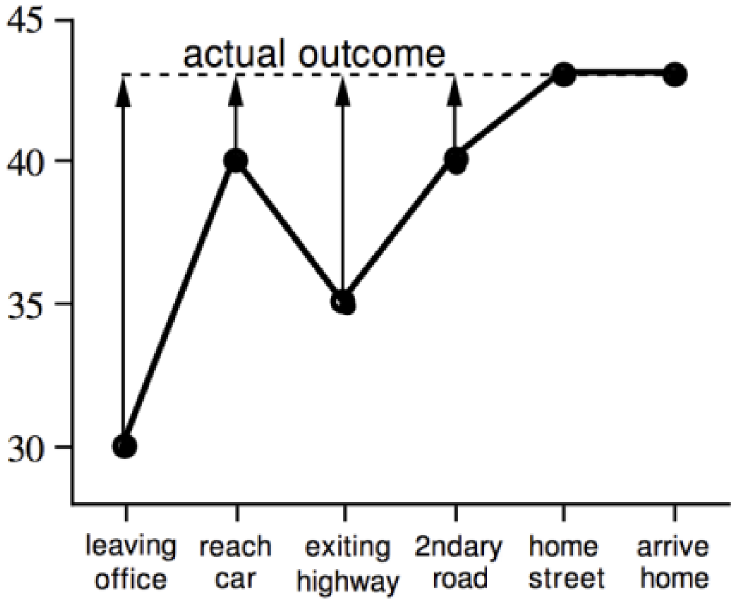
Driving Home Example

State	Elapsed Time (minutes)	Predicted Time to Go	Predicted Total Time
leaving office	0	30	30
reach car, raining	5	35	40
exit highway	20	15	35
behind truck	30	10	40
home street	40	3	43
arrive home	43	0	43

Driving Home Example – MC vs TD

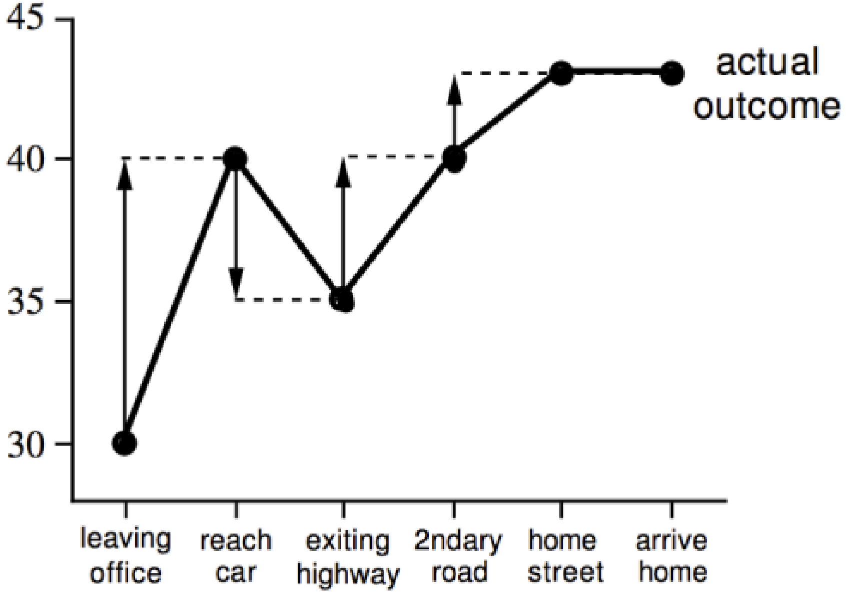
Changes recommended by MC ($\alpha = 1$)

Predicted total travel time



Situation

Changes recommended by TD ($\alpha = 1$)



Situation

Advantages and Disadvantages of MC vs. TD (I)

- ✓ 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

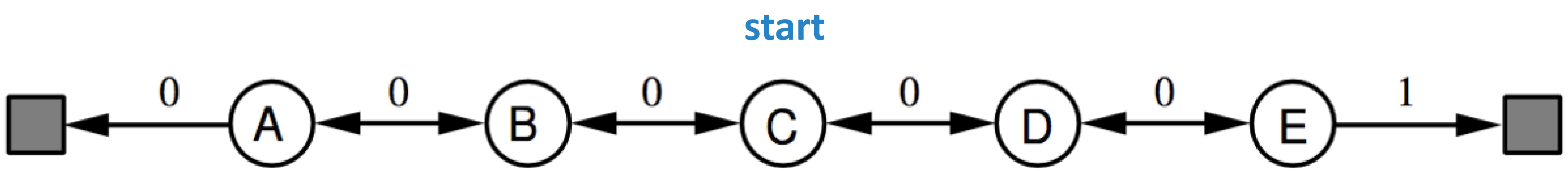
Bias-Variance Tradeoff

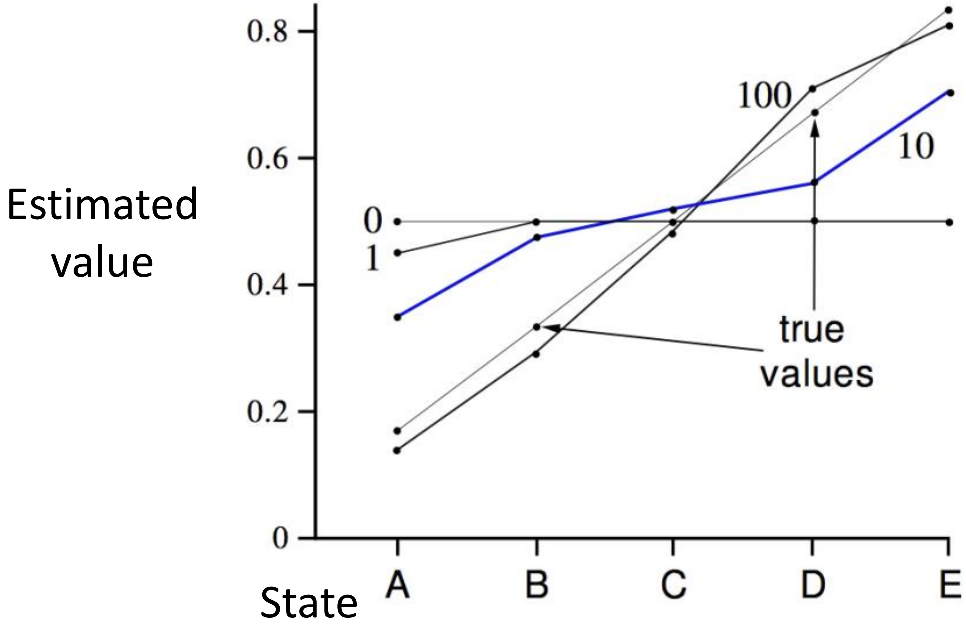
- ✓Return $G_t = R_{t+1} + \gamma R_{t+2} + \cdots, \gamma^{T-1} R_T$ is unbiased estimate of $\mathrm{v}_\pi(S_t)$
- ✓True TD target $R_{t+1} + \gamma v_\pi(S_{t+1})$ is unbiased estimate of $\mathrm{v}_\pi(S_t)$
- ✓TD target $R_{t+1} + \gamma V(S_{t+1})$ is biased estimate of $\mathrm{v}_\pi(S_t)$
- ✓TD target is much lower variance than the return:
  - ✓Return depends on many random actions, transitions, rewards
  - ✓TD target depends on one random action, transition, reward

Advantages and Disadvantages of MC vs. TD (II)

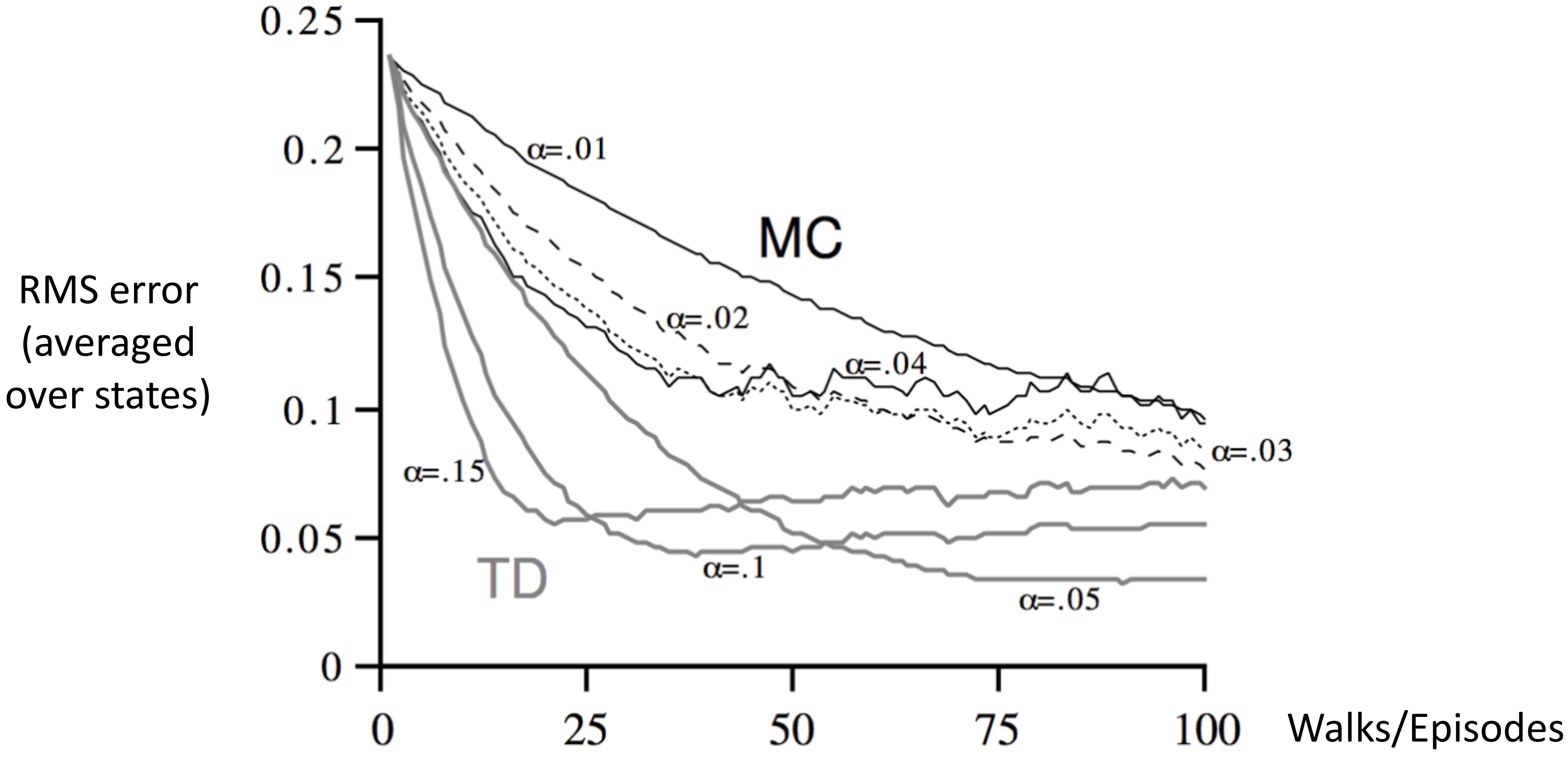
- ✓MC has high variance, zero bias
  - ✓Good convergence properties (even with function approximation)
  - ✓Not very sensitive to initial value
  - ✓Very simple to understand and use
- ✓TD has low variance, some bias
  - ✓Usually more efficient than MC
  - ✓TD(0) converges to $v_{\pi}(s)$ (but not always with function approximation)
  - ✓More sensitive to initial value

Random Walk Example





Random Walk Example – MC vs TD



Batch MC and TD

- MC and TD converge: $V(s) \rightarrow v_\pi(s)$ as experience $\rightarrow \infty$
- But what about batch solution for finite experience?

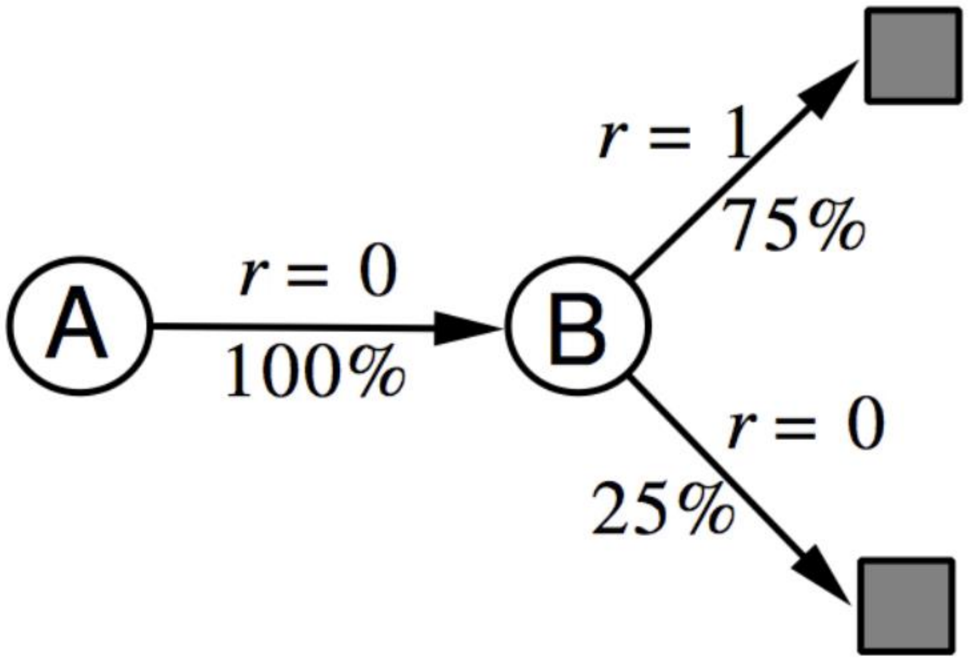
$$
\begin{gathered}
s_1^1, a_1^1, r_2^1, \ldots, s_{T_1}^1 \\
\vdots \\
s_1^K, a_1^K, r_2^K, \ldots, s_{T_K}^K
\end{gathered}
$$

  - e.g. repeatedly sample episode $k \in [1, K]$
  - Apply MC or TD(0) to episode k

A Simple Example

✓Two states A; B; no discounting; 8 episodes of experience

1. A, 0, B, 0
2. B, 1
3. B, 1
4. B, 1
5. B, 1
6. B, 1
7. B, 1
8. B, 0



✓What is V (A); V (B)?

Certainty Equivariance

- MC converges to solution with minimum mean-squared error
  - Best fit to the observed returns

$$\sum_{k=1}^{K}\sum_{t=1}^{T_k}\left(G_t^k - V(s_t^k)\right)^2$$

- TD(0) converges to solution of maximum likelihood Markov model
  - Solution to the MDP $\langle \mathcal{S}, \mathcal{A}, \boldsymbol{P}, \mathcal{R}, \gamma \rangle$ that best fits the data

$$\hat{P}_{ss'}^{a} = \frac{1}{N(s,a)}\sum_{k=1}^{K}\sum_{t=1}^{T_k}\mathbf{1}(s_t^k, a_t^k, s_{t+1}^k; s, a, s')$$

$$\hat{\mathcal{R}}_{s}^{a} = \frac{1}{N(s,a)}\sum_{k=1}^{K}\sum_{t=1}^{T_k}\mathbf{1}(s_t^k, a_t^k; s, a) r_t^k$$

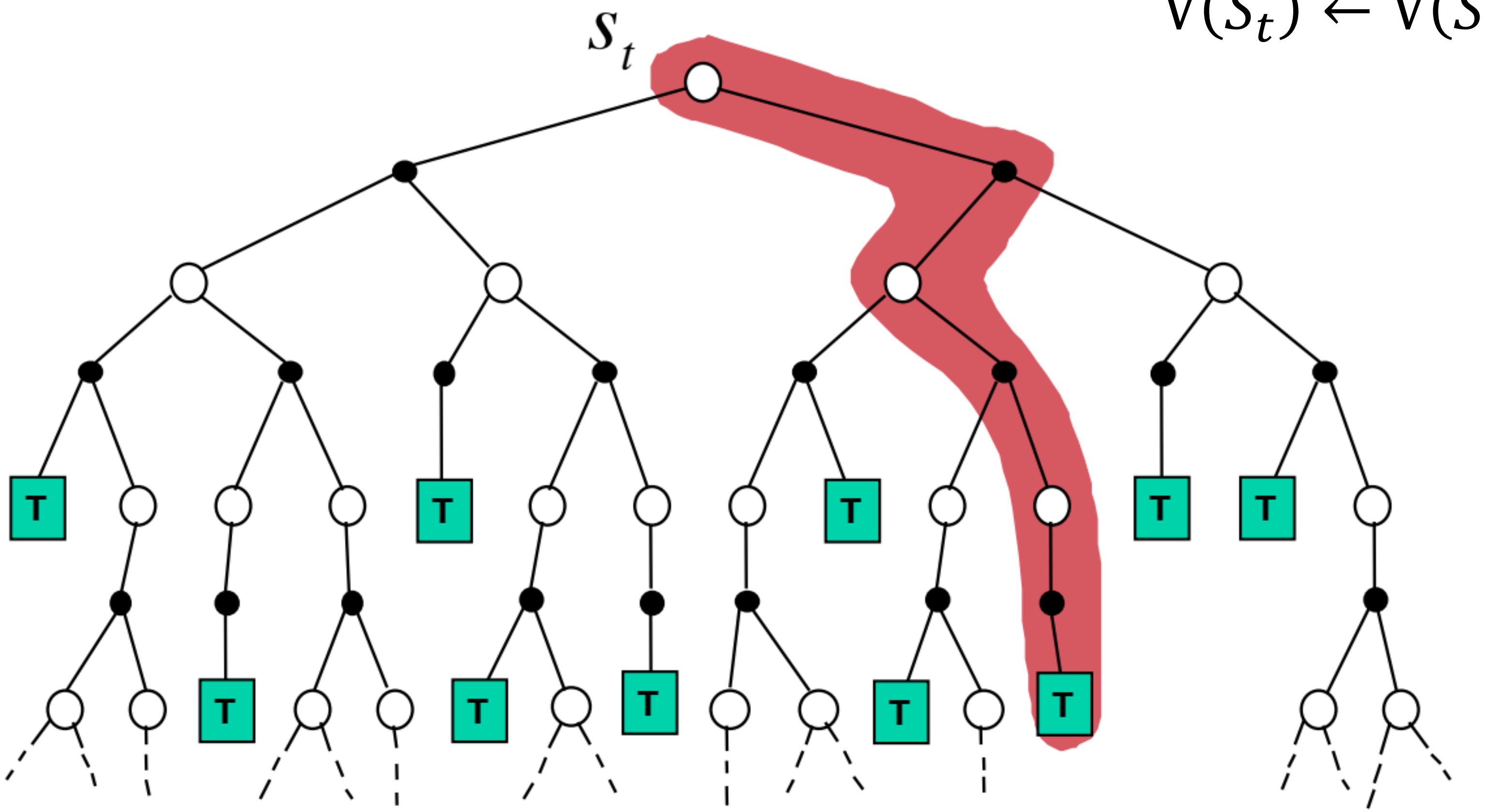
Advantages and Disadvantages of MC vs. TD (III)

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

Unified View

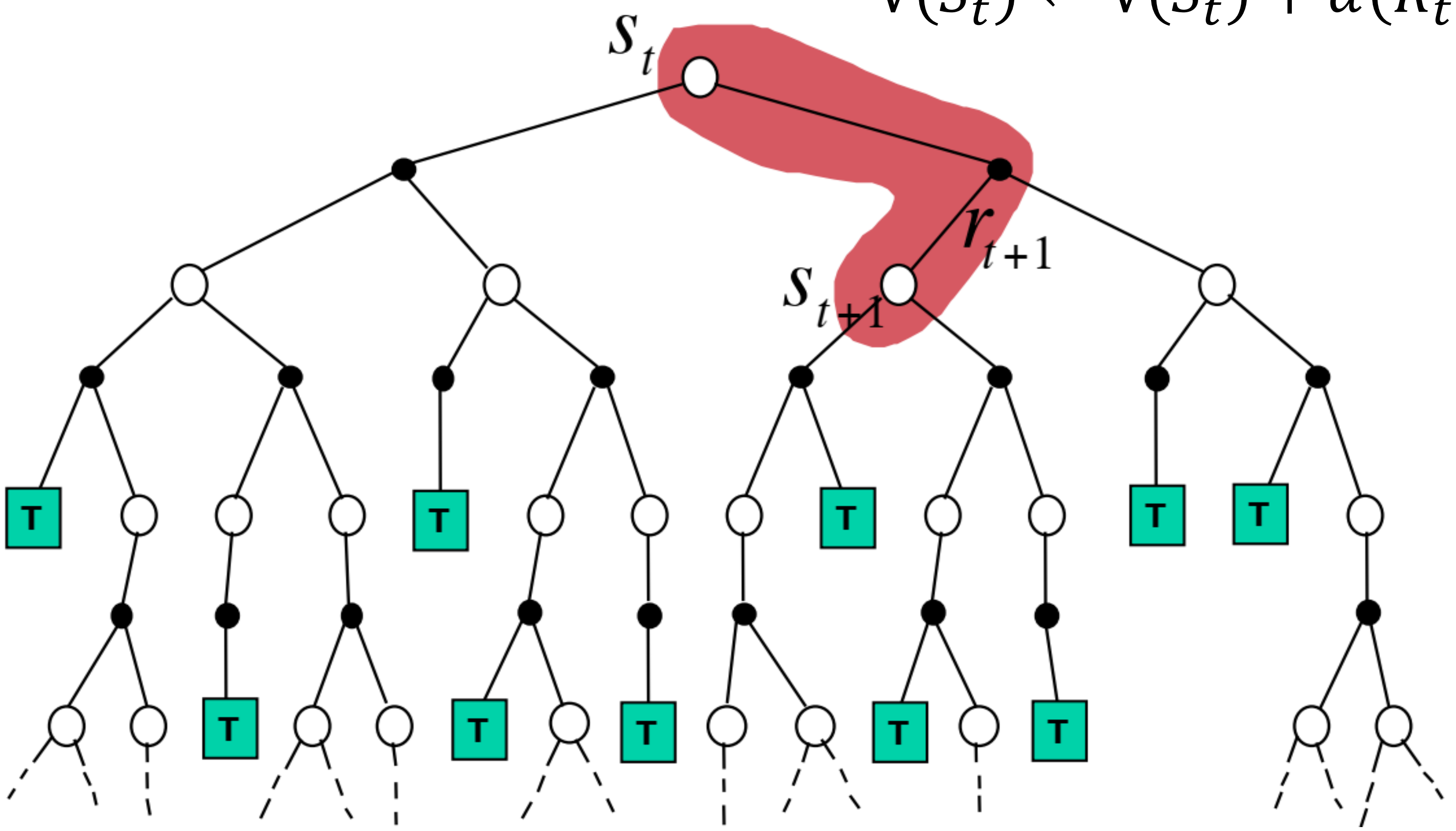
MC Update

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

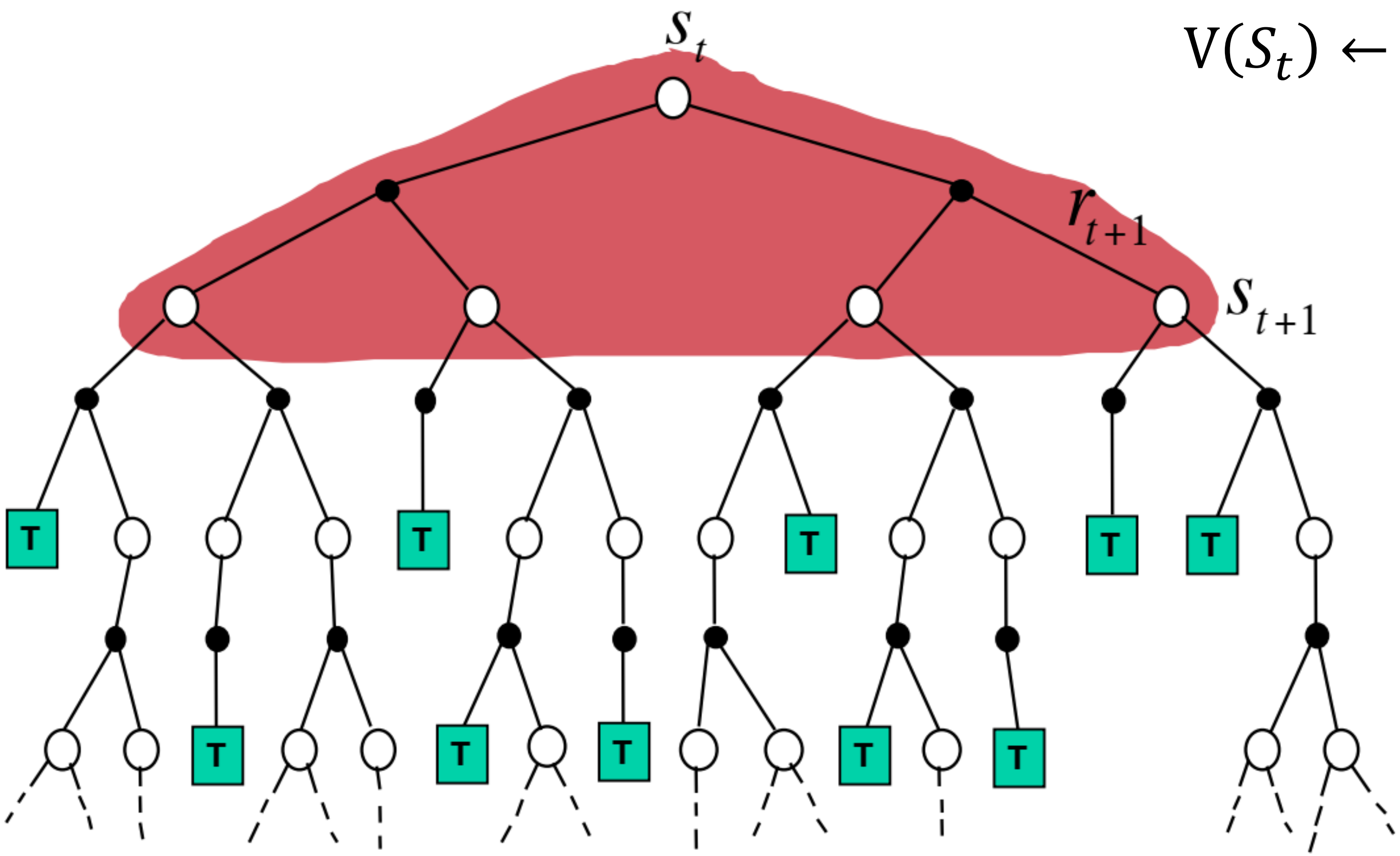


TD Update

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



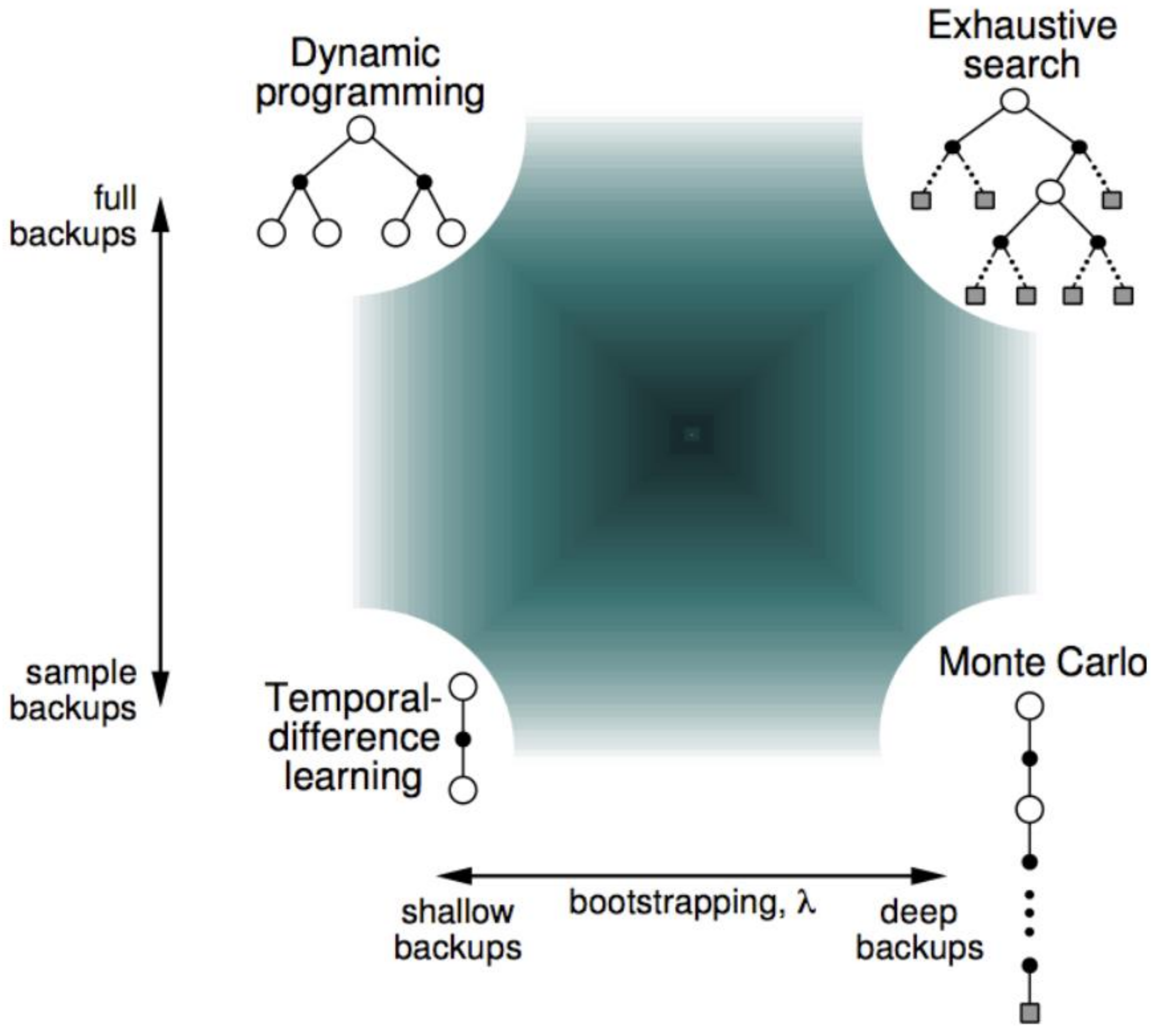
Dynamic Programming



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

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

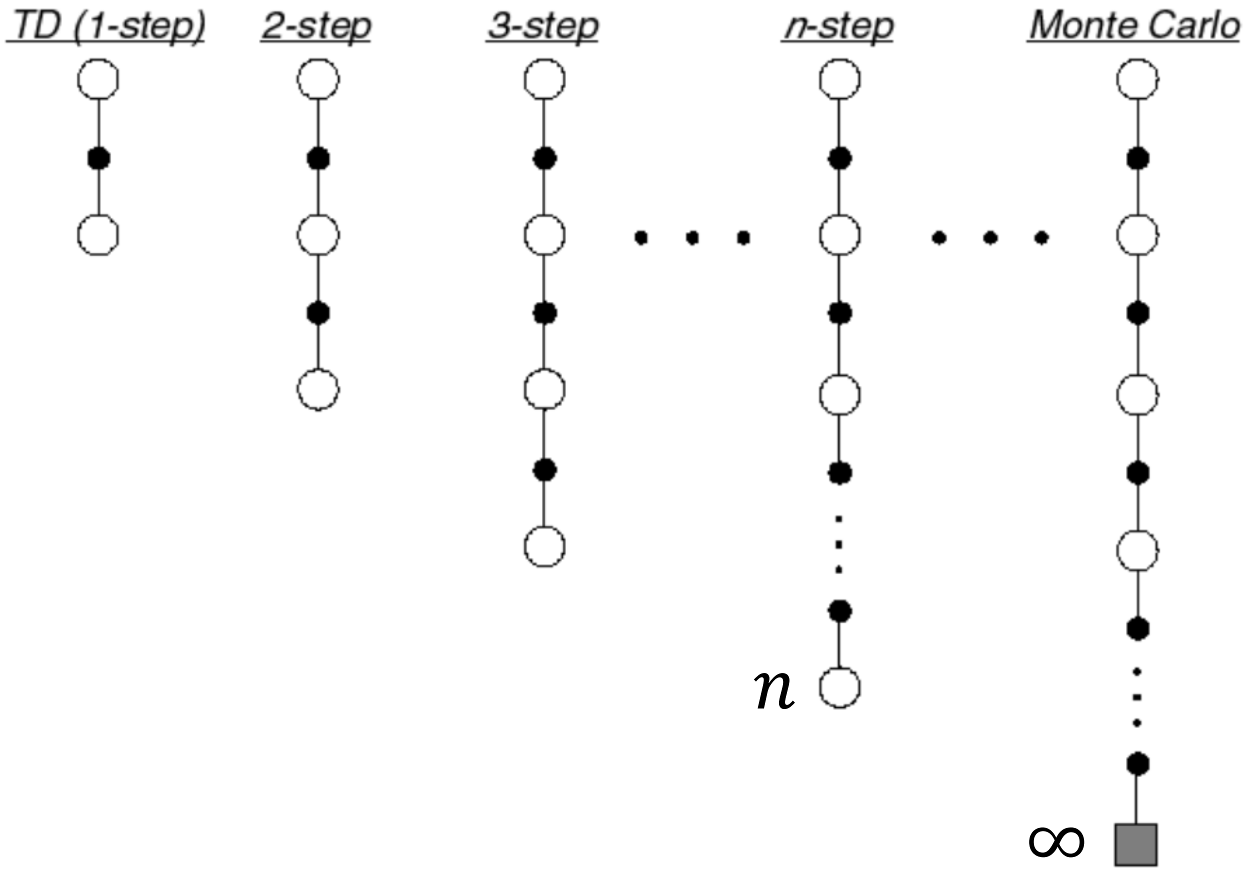


Unified View of RL

Generalizing TD

n-step Prediction

Have TD look and target *n* steps in the future



n-step Return

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

$$n = 1 \quad \text{(TD)} \quad G_t^{(1)} = R_{t+1} + \gamma V(S_{t+1})$$

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

$$\dots$$

$$n = \infty \quad \text{(MC)} \quad G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_T$$

- ✓Define the n-step return

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

- ✓Learn based on the n-step difference

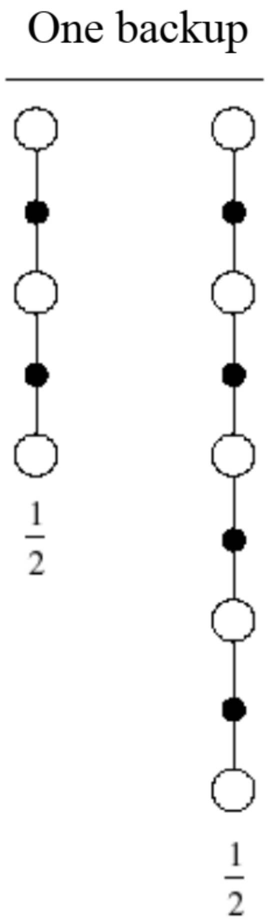
$$\mathrm{V}(S_t) \leftarrow \mathrm{V}(S_t) + \alpha \left(G_t^{(n)} - V(S_t) \right)$$

Averaging n-step Returns

- We can average n-step returns over different n
  - E.g.: Average the 2-step and 4-step returns

$$\frac{1}{2}G^{(2)} + \frac{1}{4}G^{(4)}$$

- Combines information from two different time-steps
- Can we efficiently combine information from all time-steps?



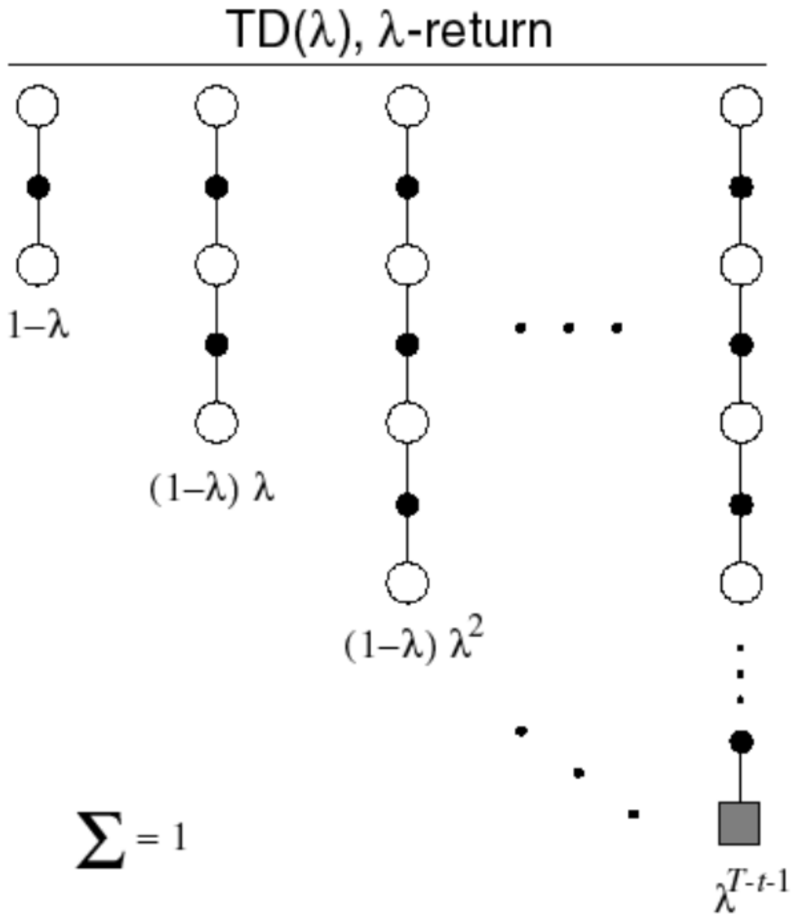
λ-returns

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

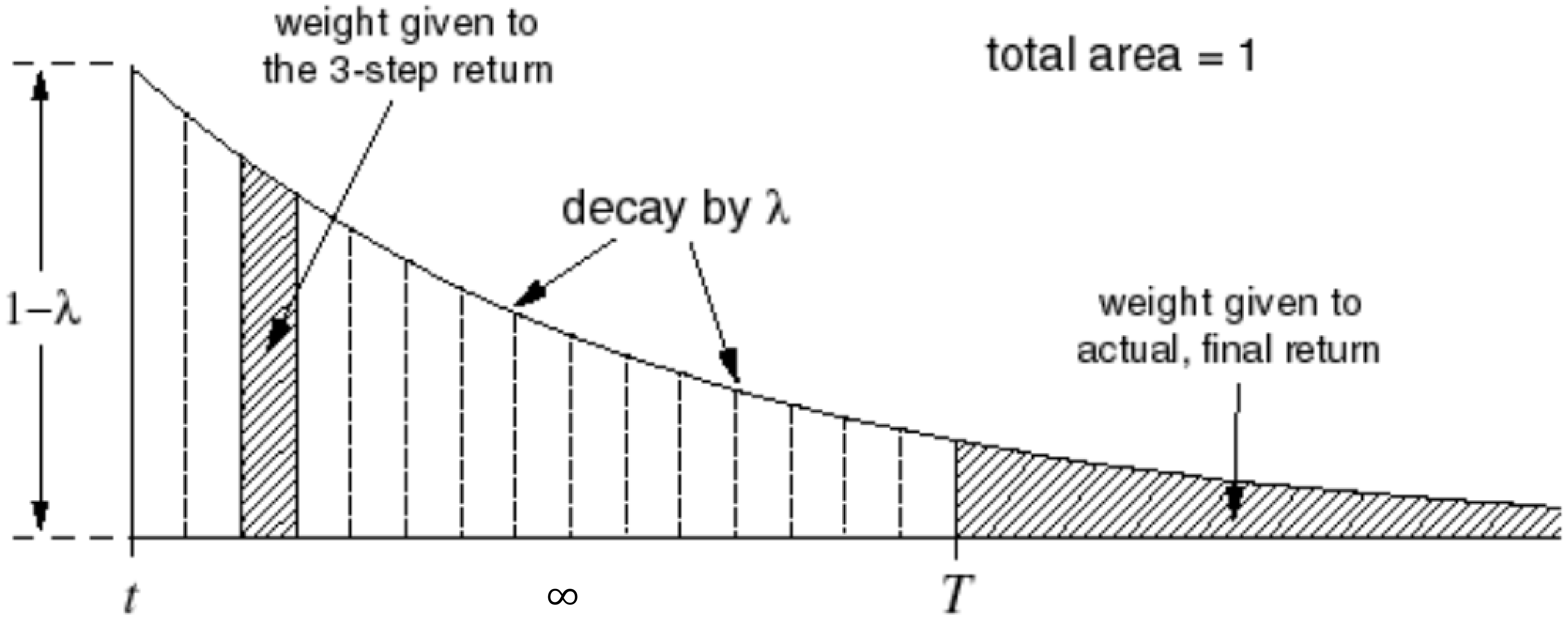
$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_t^{(n)}$$

- Update as appropriate (TD(λ))

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

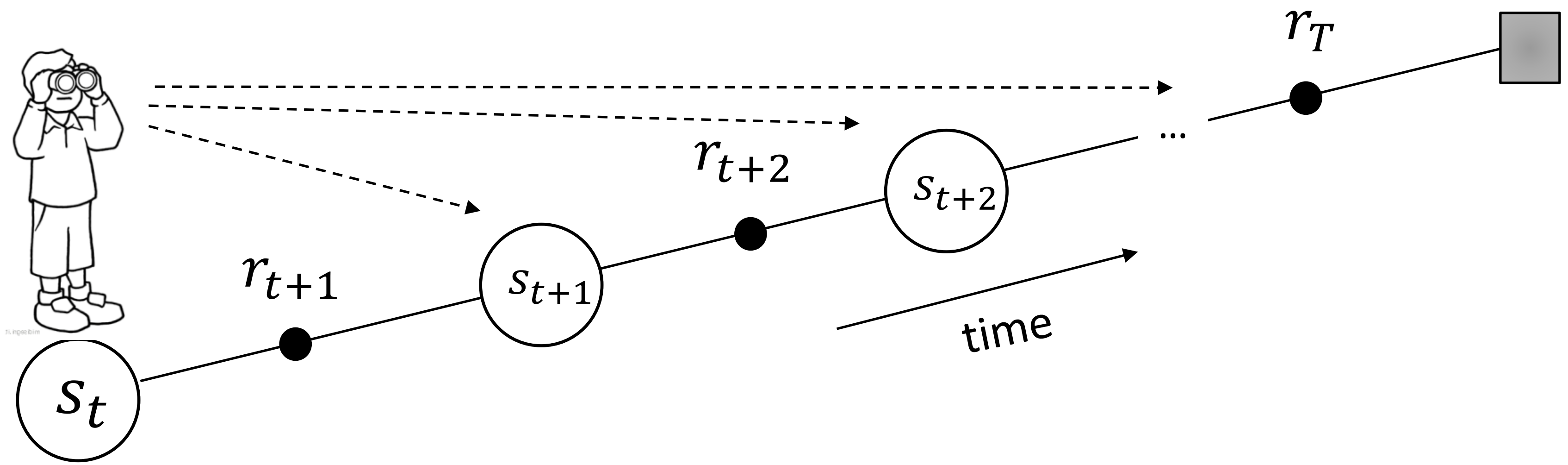


TD(λ) Weight Function



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

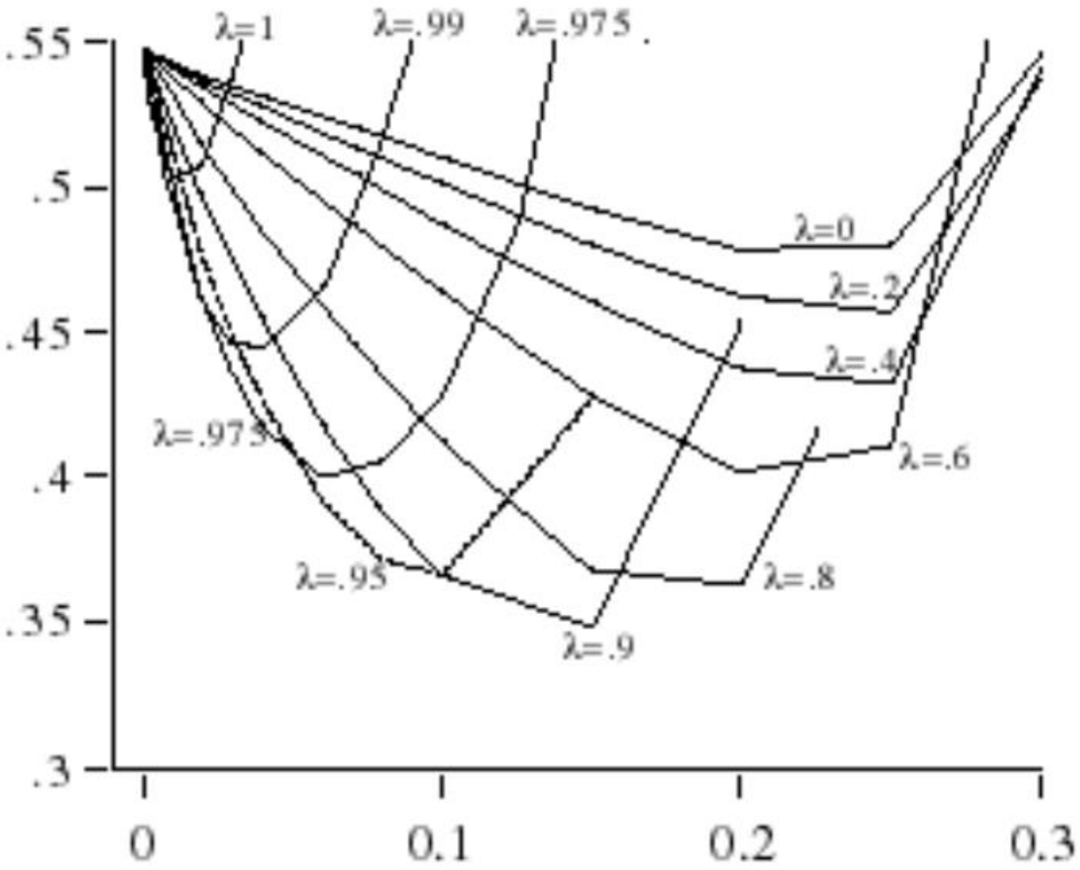
Forward View TD(λ)



- ✓ 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 TD(λ) on Large Random Walks

RMS error averaged on 10 episodes

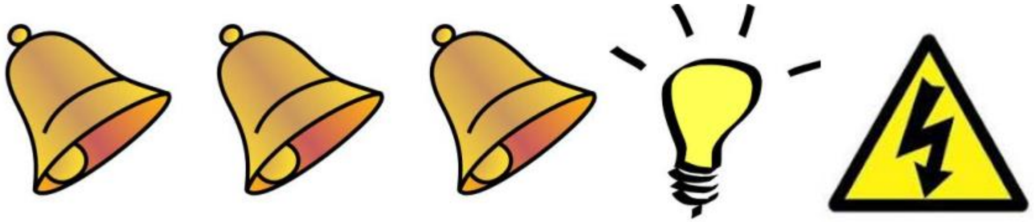


Offline λ-returns

Backward View TD(λ)

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

Eligibility Traces

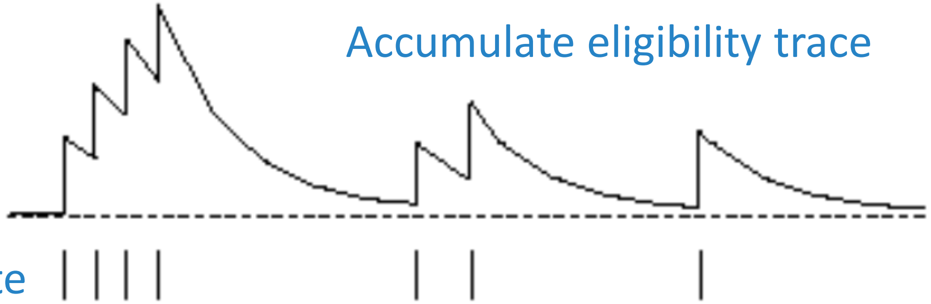


- ✓Credit assignment problem: what caused shock?
  - ✓Frequency heuristic: assign credit to most frequent states
  - ✓Recency heuristic: assign credit to most recent states
- ✓Eligibility traces combine both heuristics

$$E_0(s) = 0$$

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

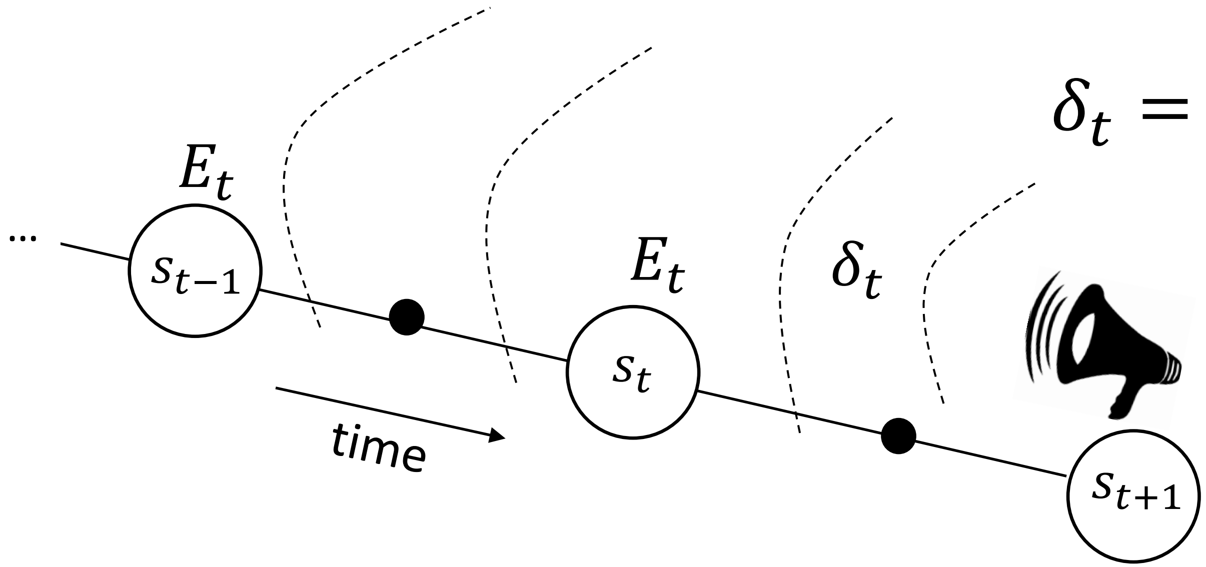
Accumulate eligibility trace



times of visit to state

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 $E_t(s)$



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

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

TD(λ) and TD($\mathbf{0}$)

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

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

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

- ✓Equivalent to TD(0) update

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

TD(λ) and MC

- ✓ When $\lambda = 1$ credit is deferred until end of episode
- ✓ Consider episodic environments with offline updates
- ✓ Over the course of an episode, total update for TD(1) is the same as total update for MC

Theorem

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

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

Telescoping in TD(1)

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

$$\ldots (proof\ in\ book\ if\ interested) \ldots$$

- ✓TD(1) is roughly equivalent to every-visit Monte-Carlo
- ✓Error is accumulated online, step-by-step
- ✓If value function is only updated offline at end of episode, then total update is the same as MC

Wrap-up

Take (stay) home messages

- ✓ Model-free prediction is value function estimation of an unknown MDP
  - ✓ Based on sample-updates
- ✓ Monte Carlo methods
  - ✓ Estimating value function by averaging sample returns
  - ✓ Only for episodic tasks (eventually terminate no matter what actions are taken)
- ✓ TD learning
  - ✓ Learn from existing (biased) estimates of future return (bootstrapping)
  - ✓ Explore the future until n-th step

Next Lecture

Model-Free Control

- ✓ Optimise the value function of an unknown MDP
  - ✓ Generalised Policy Iteration
- ✓ Monte Carlo Control
- ✓ TD learning
- ✓ On-policy Vs Off-policy

No lecture on Friday 15/05/2020 (but extra on 13/05/2020)