REINFORCEMENT LEARNING

CP8319/CPS824
Lecture 8

Instructor: Nariman Farsad

* Some of the slides in this deck are adopted from courses offered David Silver, Emma Brunskill, and Sergey Levine.

Today’s Agenda

1. **Finish Monte Carlo Policy Evaluation**

2. Temporal Difference Policy Evaluation

What we have learned up to now?

So far we have solved a *known* MDP, i.e., dynamics and the reward function are known

Moving forward:

- Estimate the value function of an *unknown* MDP

- Optimize the value function of an *unknown* MDP

First Visit MC (On) Policy Evaluation

Goal: estimate $v^\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]\ \forall s \in \mathcal{S}$

Initialize $N(s) = 0, G(s) = 0\ \forall s \in \mathcal{S}$

Loop:

- Sample episode i: $s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \dots, s_{i,T_i}$
- 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 i-th 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 visits: $N(s) = N(s) + 1$
    - Increment total return $G(s) = G(s) + G_{i,t}$
    - Update estimate $v^\pi(s) = G(s)/N(s)$

Every-Visit MC (On) Policy Evaluation

Goal: estimate $v^{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s] \; \forall s \in \mathcal{S}$

Initialize $N(s) = 0, G(s) = 0 \; \forall s \in \mathcal{S}$

Loop:

- Sample episode i: $s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \dots, s_{i,T_i}$
- 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 i-th 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 visits: $N(s) = N(s) + 1$
    - Increment total return $G(s) = G(s) + G_{i,t}$
    - Update estimate $v^{\pi}(s) = G(s)/N(s)$

Incremental Mean Calculation

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

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

Incremental MC (On) Policy Evaluation

Initialize $N(s) = 0, G(s) = 0 \; \forall s \in \mathcal{S}$

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 i-th 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 visits: $N(s) = N(s) + 1$
        - Update estimate $v^\pi(s) = v^\pi(s) + \frac{1}{N(s)}(G_{i,t} - v^\pi(s))$

Incremental MC (On) Policy Evaluation

Initialize $N(s) = 0, G(s) = 0 \; \forall s \in \mathcal{S}$

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 i-th 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 visits: $N(s) = N(s) + 1$
    - Update estimate $v^\pi(s) = v^\pi(s) + \alpha(G_{i,t} - v^\pi(s))$

$\alpha = \frac{1}{N(s)}$: Identical to every visit MC

$\alpha > \frac{1}{N(s)}$: forget older data, helpful for non-stationary domains

Bias, Variance, and MSE

- Consider a statistical model that is parameterized by θ and that determines a probability distribution over observed data $P(x|\theta)$
- Consider a statistic $\hat{\theta}$ that provides an estimate of θ and is a function of observed data x
  - E.g. for a Gaussian distribution with known variance, the average of a set of i.i.d data points is an estimate of the mean of the Gaussian
- Definition: the bias of an estimator $\hat{\theta}$ is:

$$Bias_\theta(\hat{\theta}) = \mathbb{E}_{x|\theta}[\hat{\theta}] - \theta$$

- Definition: the variance of an estimator $\hat{\theta}$ is:

$$Var(\hat{\theta}) = \mathbb{E}_{x|\theta}[(\hat{\theta} - \mathbb{E}[\hat{\theta}])^2]$$

- Definition: mean squared error (MSE) of an estimator $\hat{\theta}$ is:

$$MSE(\hat{\theta}) = Var(\hat{\theta}) + Bias_\theta(\hat{\theta})^2$$

Bias and Variance of MC Policy Evaluation

First Visit MC:

- v^π estimator is an *unbiased* estimator of true $\mathbb{E}_\pi[G_t|S_t = s]$
- By law of large numbers, as $N(s) \to \infty, v^\pi \to \mathbb{E}_\pi[G_t|S_t = s]$

Every Visit MC:

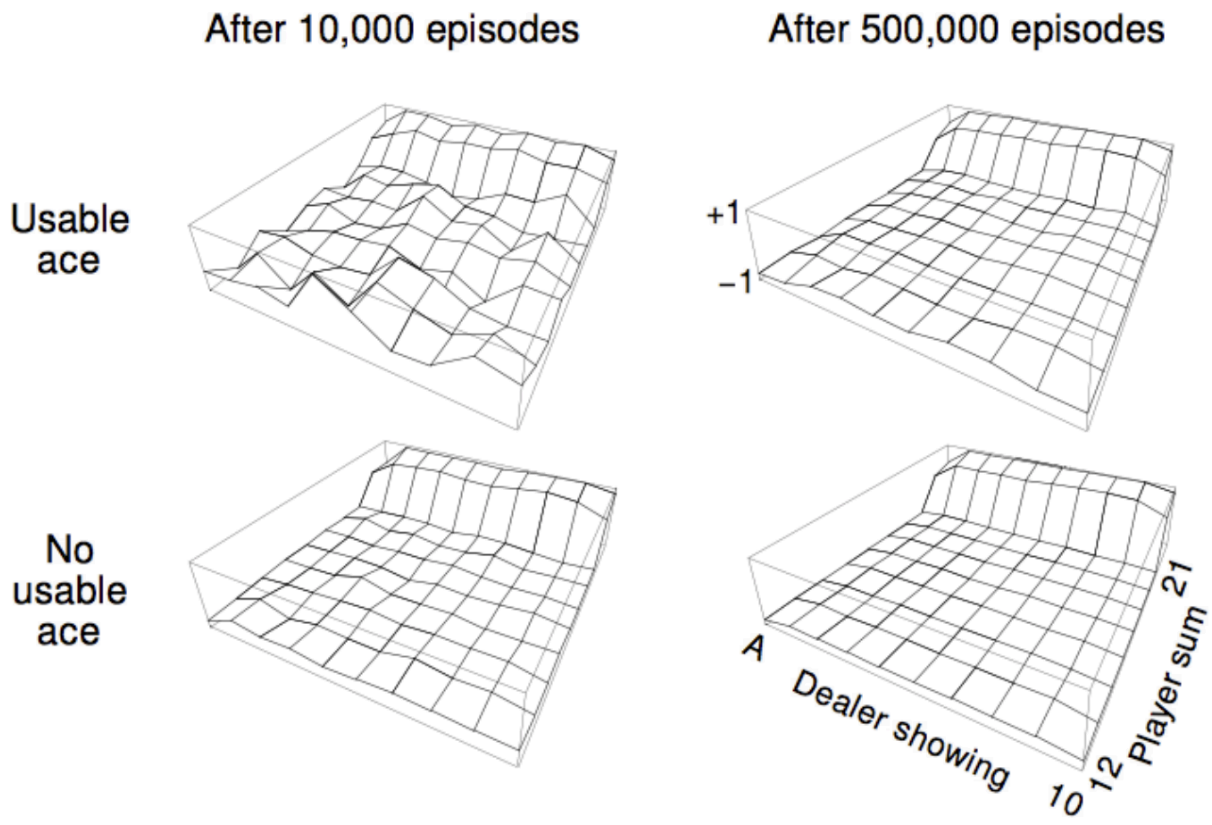
- v^π estimator is a *biased* estimator, but it is a *consistent* estimator
- *consistent* estimator: As $N(s) \to \infty$, v^π estimate can get *arbitrarily close* to true value of v^π
- Often has better MSE

Both every visit and first are high variance estimators of v^π

Example: Blackjack

- States (200 of them):
  - Current sum (12-21)
  - Dealer's showing card (ace-10)
  - Do I have a "useable" ace? (yes-no)
- Action hold: Stop receiving cards (and terminate)
- Action hit: Take another card (no replacement)
- Reward for hold:
  - +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 hit:
  - -1 if sum of cards > 21 (and terminate)
  - 0 otherwise
- Transitions: automatically hit if sum of cards < 12

Example: Blackjack



Policy: hold if sum of cards ≥ 20, otherwise hit

MC Summary

- Generally high variance estimator
  - Reducing variance can require a lot of data
  - In cases where data is very hard or expensive to acquire, or the stakes are high, MC may be impractical
- Requires episodic settings
  - Episode must end before data from episode can be used to update v

Today’s Agenda

1. Finish Monte Carlo Policy Evaluation

2. **Temporal Difference Policy Evaluation**

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* (i.e., using estimates to update value function)
- *Bootstrapping:* TD updates a guess towards a guess (i.e., using estimates of value function to re-estimate the value function!)
- "If one had to identify one idea as central and novel to reinforcement learning, it would undoubtedly be temporal-difference (TD) learning." – Sutton and Barto 2017

TD Policy Evaluation

- Aim: estimate $v^\pi(s)$ given episodes generated under policy π
- $G_t = rt + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots$ under policy π
- $v^\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$
- Recall Bellman operator (if know MDP models):

$$\mathfrak{B}^\pi v(s) = R(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} P_{s,s'}^{\pi(s)} v(s')$$

- In incremental every-visit MC, update estimate using:

$$v^\pi(s_t) = v^\pi(s_t) + \alpha(G_{i,t} - v^\pi(s_t))$$

- Insight: have an estimate of v, use to estimate expected return

$$v^\pi(s_t) = v^\pi(s_t) + \alpha([r_t + \gamma v^\pi(s_{t+1})] - v^\pi(s_t))$$

TD(0) Policy Evaluation

- Aim: estimate $v^\pi(s)$ given episodes generated under policy π
- $G_t = rt + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots$ under policy π
- $v^\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$
- Simplest temporal-difference learning algorithm: TD(0)
  - Update value $v^\pi(s_t)$ toward estimated return $r_t + \gamma v^\pi(s_{t+1})$

$$v^\pi(s_t) = v^\pi(s_t) + \alpha([r_t + \gamma v^\pi(s_{t+1})] - v^\pi(s_t))$$

- $r_t + \gamma v^\pi(s_{t+1})$ is called the ***TD target***
- $\delta_t = r_t + \gamma v^\pi(s_{t+1}) - v^\pi(s_t)$ is called the ***TD error***
- Can immediately update value estimate after (s, a, r, s') tuple
- Don't need episodic setting

TD(0) Policy Evaluation Algorithm

Input: α
Initialize $v^{\pi}(s) = 0, \forall s \in \mathcal{S}$
Loop

- Sample tuple (s_t, a_t, r_t, s_{t+1})
- $v^{\pi}(s_t) = v^{\pi}(s_t) + \alpha([r_t + \gamma v^{\pi}(s_{t+1})] - v^{\pi}(s_t))$

TD(0) Policy Evaluation Algorithm Example

Input: α

Initialize $v^\pi(s) = 0, \forall s \in \mathcal{S}$

Loop

- Sample tuple (s_t, a_t, r_t, s_{t+1})
- $v^\pi(s_t) = v^\pi(s_t) + \alpha([r_t + \gamma v^\pi(s_{t+1})] - v^\pi(s_t))$

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

- R = [1 0 0 0 0 0 +10] for any action
- $\pi(s) = a_1 \; \forall s, \gamma = 1$. Any action from s1 and s7 terminates episode
- Sample episode = $(s_3, a_1, 0, s_2, a_1, 0, s_2, a_1, 0, s_1, a_1, 1,$ terminal)
- First/every visit MC estimate of v of each state? [1,1,1,0,0,0,0]
- TD estimate of all states (init at 0) with $\alpha = 1$?

TD vs MC

- 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 from incomplete sequences
  - MC can only learn from complete sequences
  - TD works in continuing (non-terminating) environments
  - MC only works for episodic (terminating) environments

TD vs MC: Bias and Variance

- MC has high variance, zero bias (first-visit)
  - Good convergence properties (even with function approximation)
    - *Function approximation*: used in infinite state MDPs. We will learn about it later
  - Not very sensitive to initial values used in the initialization
  - Very simple to understand and use
- TD has low variance, some bias
  - Usually more efficient than MC
  - TD(0) converges (but not always with function approximation)
  - More sensitive to initial values used in the initialization

TD vs MC: Markov vs Non-Markov

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

Batch MC and TD

- *Batch (Offline)* solution for finite dataset
- Given set of K episodes
  - episode 1: $s_{1,1}, a_{1,1}, r_{1,1}, s_{1,2}, a_{1,2}, r_{1,2}, \dots, s_{1,T_1}$
  - episode 2: $s_{2,1}, a_{2,1}, r_{2,1}, s_{2,2}, a_{2,2}, r_{2,2}, \dots, s_{2,T_2}$
  - …
  - episode K: $s_{K,1}, a_{K,1}, r_{K,1}, s_{K,2}, a_{K,2}, r_{K,2}, \dots, s_{K,T_2}$
- Repeatedly sample an episode from 1 to K
- Apply MC or TD(0) to the sampled episode
- What do MC and TD(0) converge to?

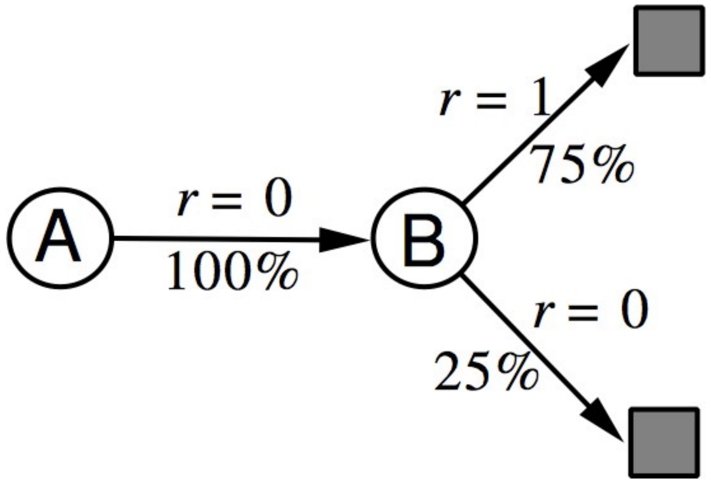
AB Example (Ex. 6.4, Sutton & Barto, 2018)

- Two states A, B with $\gamma = 1$ and a single action (i.e. action is irrelevant). Given 8 episodes of experience:
  - A,0,B,0
  - B,1 (observed 6 times)
  - B,0
- What is $v(A), v(B)$ if we run TD and MC over these specific episodes many times?

AB Example (Ex. 6.4, Sutton & Barto, 2018)

- Two states A, B with $\gamma = 1$ and a single action (i.e. action is irrelevant). Given 8 episodes of experience:
  - A,0,B,0
  - B,1 (observed 6 times)
  - B,0
- What is $v(A), v(B)$ if we run TD and MC over these specific episodes many times?

  - TD Learns the MDP dynamics
  - Uses the learned dynamics to evaluate v



What Do Batch MC and TD Converge To?

- Monte Carlo in batch setting converges to min MSE (mean squared error)

$$\mathcal{L} = \sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(G_{k,t} - v(s_{k,t}) \right)^2$$

  - Minimize the loss $\mathcal{L}$ with respect to observed returns

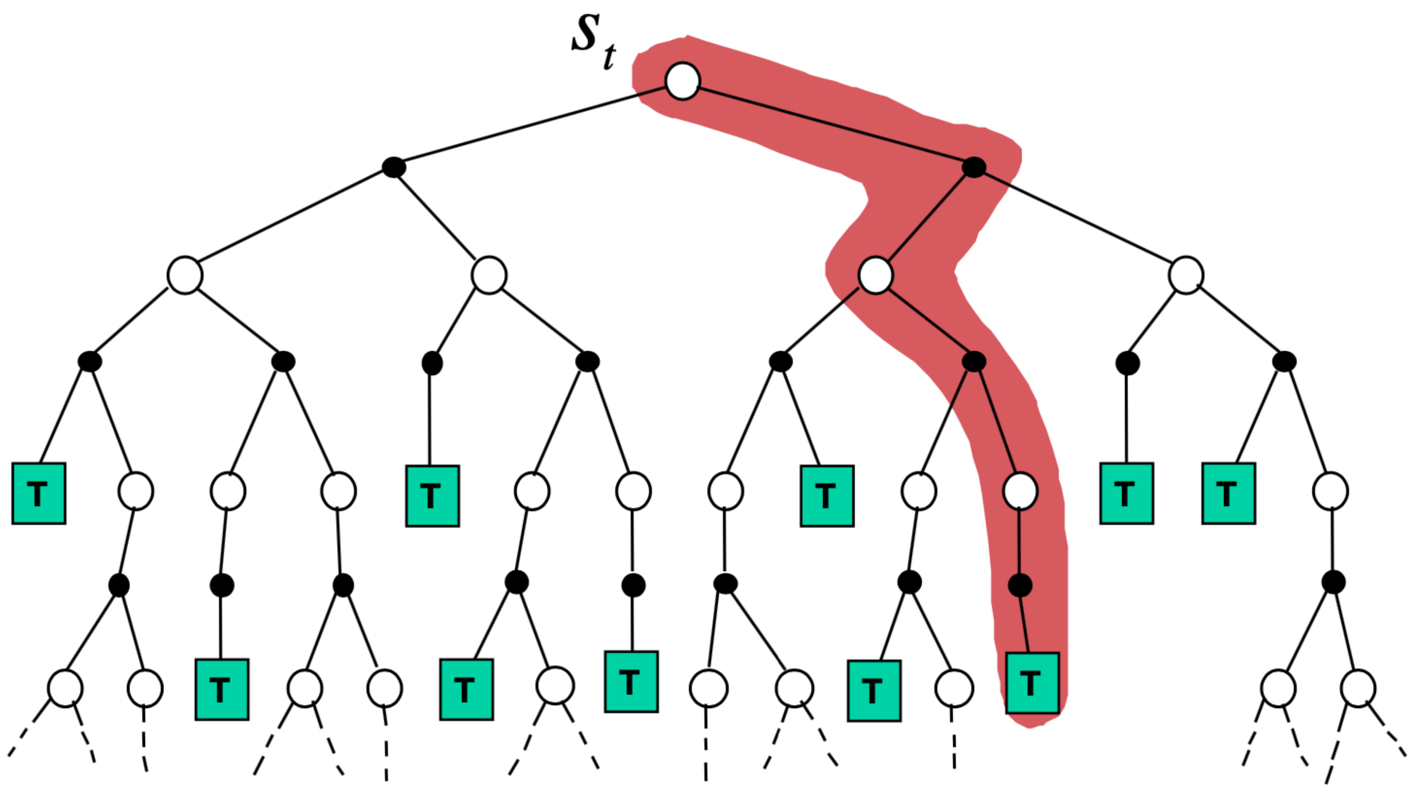
- TD(0) converges to solution of max likelihood Markov model
  - Solution to the MDP $\langle \mathcal{S}, \mathcal{A}, \widehat{\mathbf{P}}, \hat{R}, \gamma \rangle$ that best fits the data

$$\hat{P}_{s,s'}^{a} = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{T_k} \mathbf{1}\left(s_{k,t}, a_{k,t}, s_{k,t+1} = s, a, s'\right)$$

$$\hat{R}(s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{T_k} \mathbf{1}\left(s_{k,t}, a_{k,t} = s, a\right)$$

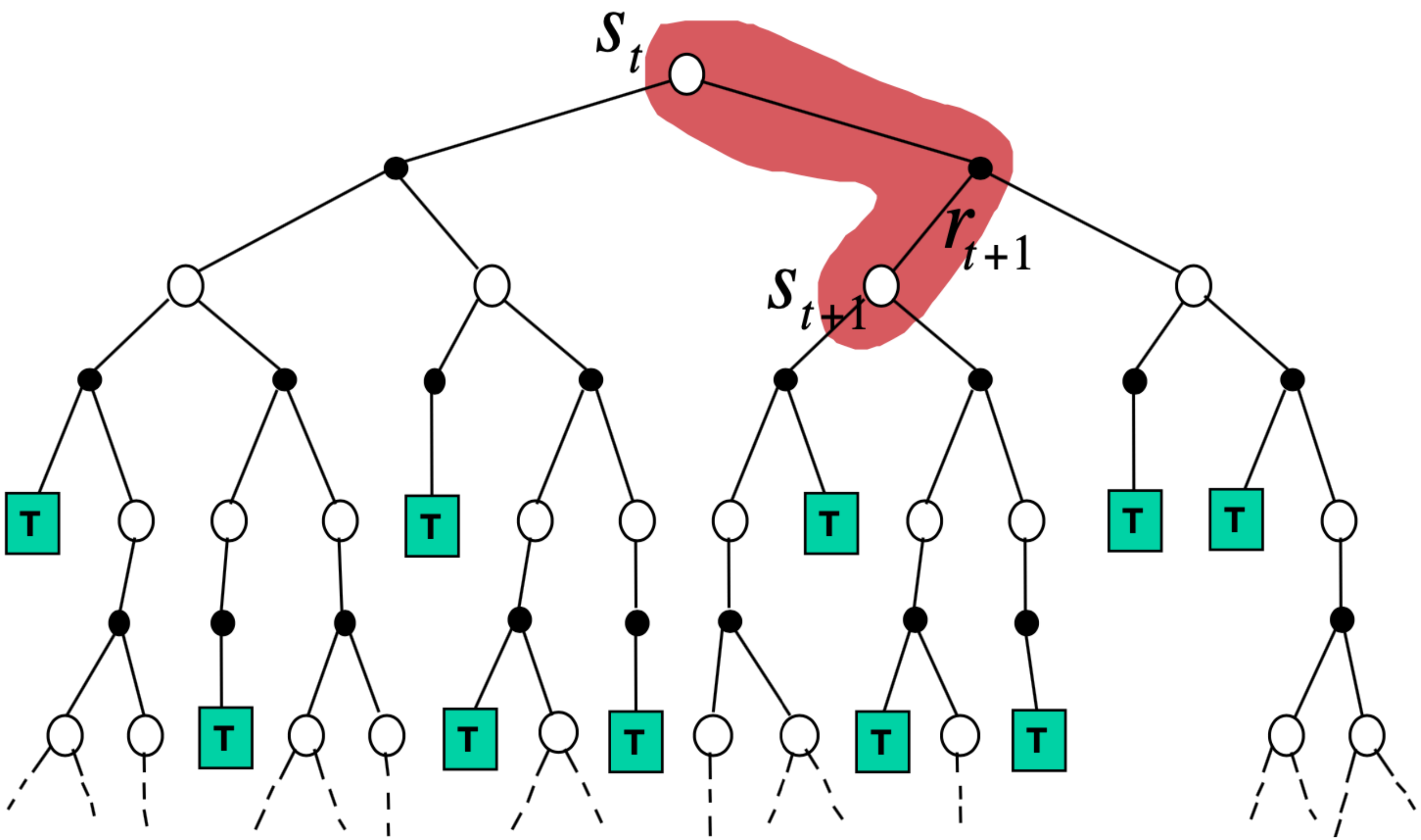
Monte Carlo Backup

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



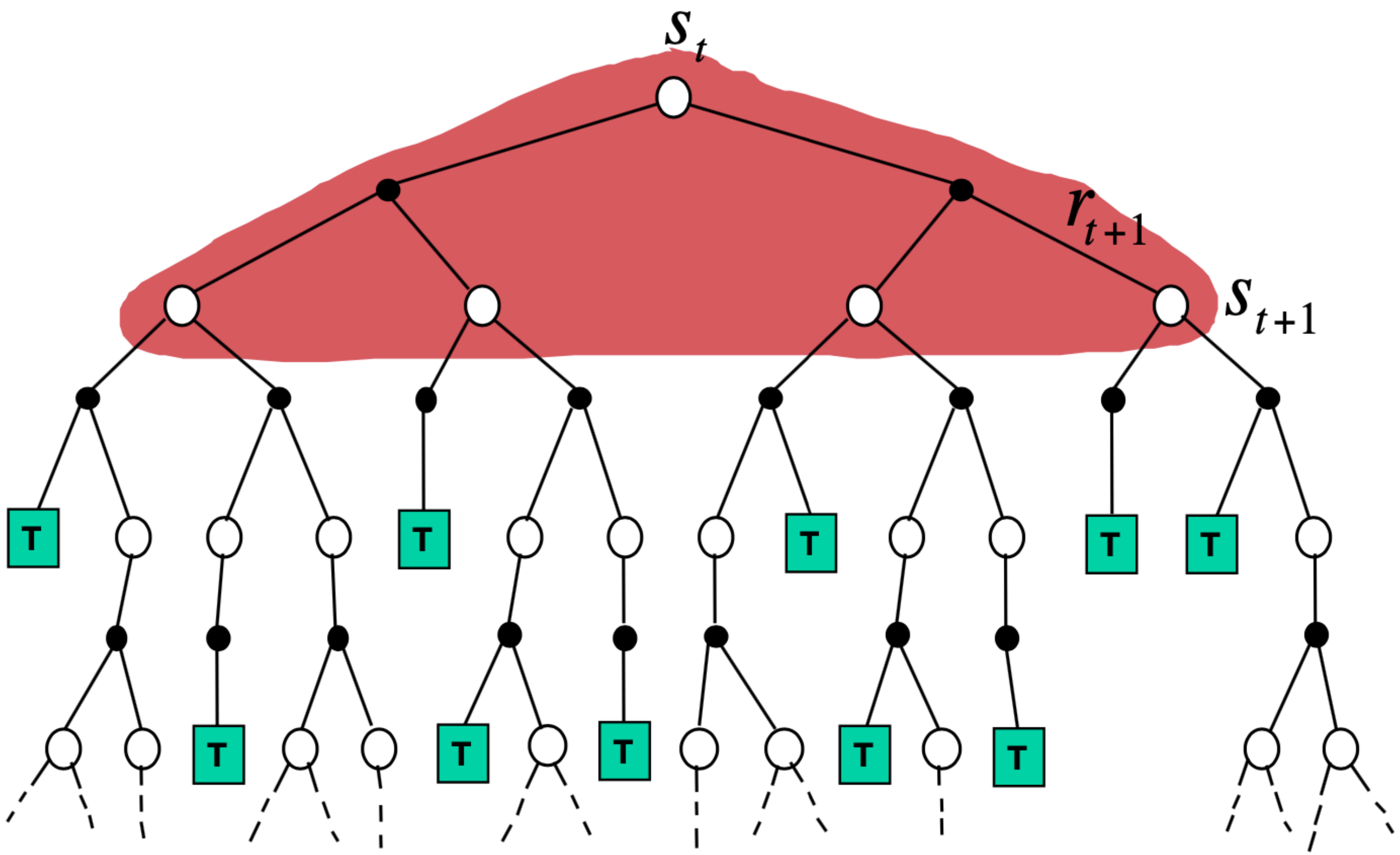
TD Backup

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



Dynamic Programming Backup

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

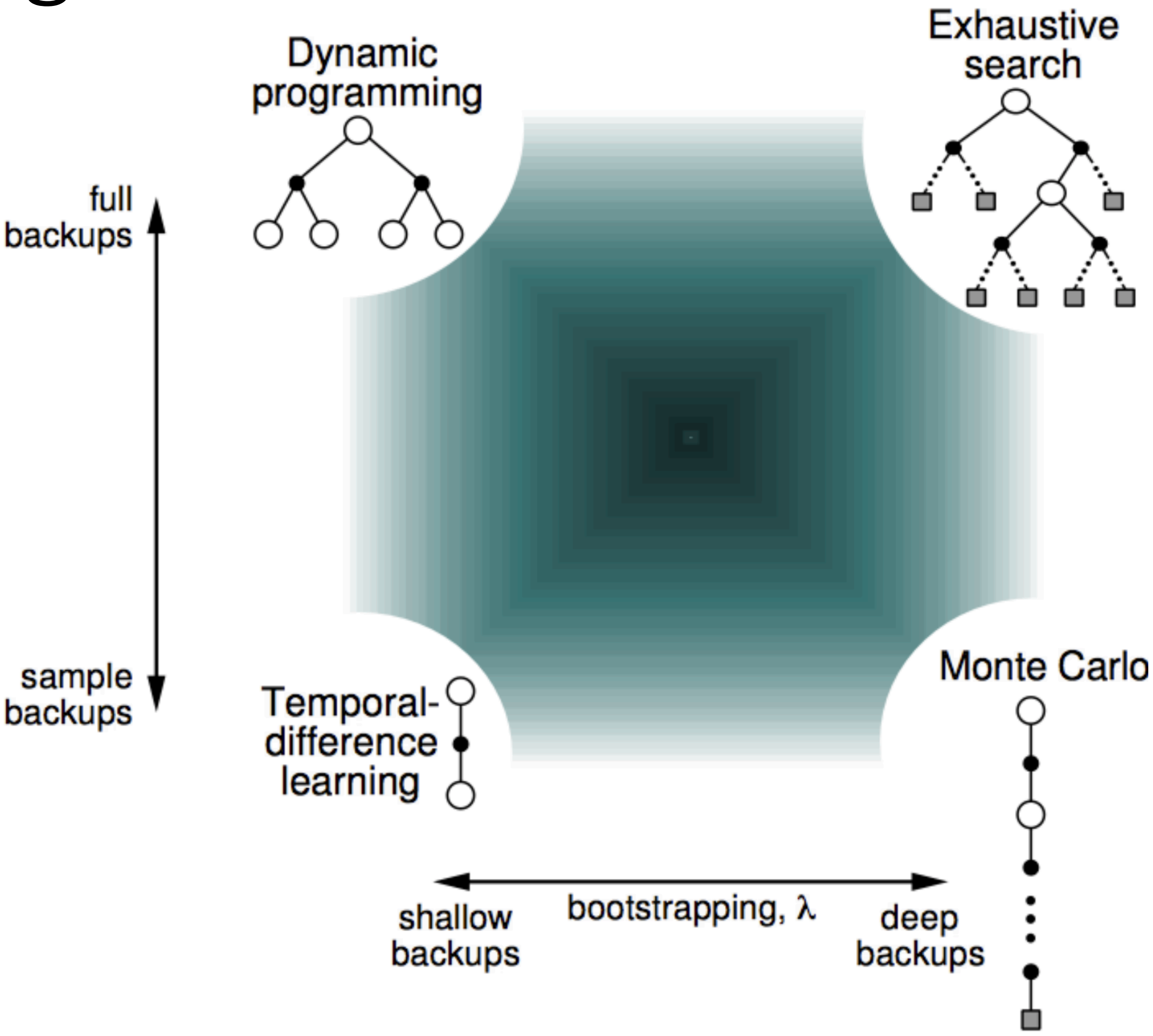


Dynamic Programming:

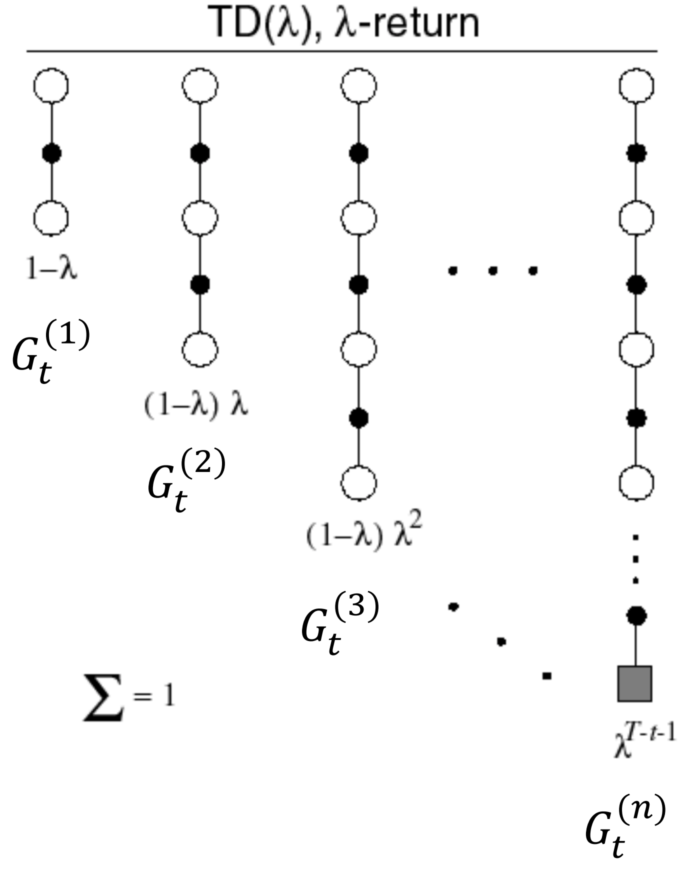
- Value iteration
- Policy iteration

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



TD(λ)



- n-step returns $G_t^{(n)}$

$$G_t^{(1)} = r_t + \gamma v(s_{t+1})$$
$$G_t^{(2)} = r_t + \gamma r_{t+1} + \gamma^2\, v(s_{t+2})$$
$$G_t^{(3)} = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 v(s_{t+3})$$
$$\vdots$$

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

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

- TD(λ): Update value function using (forward-view TD(λ)):

$$v(s_t) = v(s_t) + \alpha\left(G_t^{\lambda} - v(s_t)\right)$$

Summary of Policy Evaluation Methods

	DP	MC	TD
Usable w/no models of domain			
Handles continuing (non-episodic) setting			
Assumes Markov process			
Converges to true value in limit[1]			
Unbiased estimate of value			

- DP = Dynamic Programming (i.e., policy iteration or value iteration)
- MC = Monte Carlo
- TD = Temporal Difference

[1]For tabular representations of value function (i.e., finite number of states).