Lecture 3: Policy Evaluation Without Knowing How the World Works / Model Free Policy Evaluation

CS234: RL

Emma Brunskill

Winter 2018

Material builds on structure from David SIlver's Lecture 4: Model-Free Prediction: http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html . Other resources: Sutton and Barto Jan 1 2018 draft (http://incompleteideas.net/book/the-book-2nd.html) Chapter/Sections: 5.1; 5.5; 6.1-6.3

Class Structure

- Last Time:
  - Markov reward / decision processes
  - Policy evaluation & control when have true model (of how the world works)
- **Today:**
  - **Policy evaluation when don’t have a model of how the world works**
- Next time:
  - Control when don’t have a model of how the world works

This Lecture: Policy Evaluation

- **Estimating the expected return of a particular policy if don't have access to true MDP models**

- Dynamic programming
- Monte Carlo policy evaluation
  - Policy evaluation when don't have a model of how the world work
    - Given on-policy samples
    - Given off-policy samples
- Temporal Difference (TD)
- Metrics to evaluate and compare algorithms

Recall

- **Definition of return G_t for a MDP under policy π**:
  - Discounted sum of rewards from time step t to horizon when following policy π(a|s)
  - $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$
- **Definition of state value function $V^\pi(s)$ for policy π**:
  - Expected return from starting in state s under policy **π**
  - $V^\pi(s) = \mathbb{E}_\pi [G_t | s_t = s] = \mathbb{E}_\pi [r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots | s_t = s]$
- **Definition of state-action value function $Q^\pi(s,a)$ for policy π**:
  - Expected return from starting in state s, taking action a, and then following policy **π**
  - $Q^\pi(s,a) = \mathbb{E}_\pi [G_t | s_t = s, a_t = a] = \mathbb{E}_\pi [r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots | s_t = s, a_t = a]$

Dynamic Programming for Policy Evaluation

- Initialize $V_0(s) = 0$ for all s
- For k=1 until convergence
  - For all s in S:

$$V_k^\pi(s) = R^\pi(s) + \gamma \sum_{s' \in S} P^\pi(s'|s) V_{k-1}^\pi(s')$$

$$\begin{aligned} R^\pi(s) &= \sum_{a \in A} \pi(a|s) R(s,a) \\ P^\pi(s'|s) &= \sum_{a \in A} \pi(a|s) P(s'|s,a) \end{aligned}$$

Dynamic Programming for Policy Evaluation

- Initialize $V_0(s) = 0$ for all s
- For k=1 until convergence
  - For all s in S:

Bellman backup for a particular policy

$$V_k^\pi(s) = B^\pi V_{k-1}^\pi(s) = R^\pi(s) + \gamma \sum_{s' \in S} P^\pi(s'|s) V_{k-1}^\pi(s')$$

$$\begin{aligned} R^\pi(s) &= \sum_{a \in A} \pi(a|s) R(s,a) \\ P^\pi(s'|s) &= \sum_{a \in A} \pi(a|s) P(s'|s,a) \end{aligned}$$

Dynamic Programming for Policy π Value Evaluation

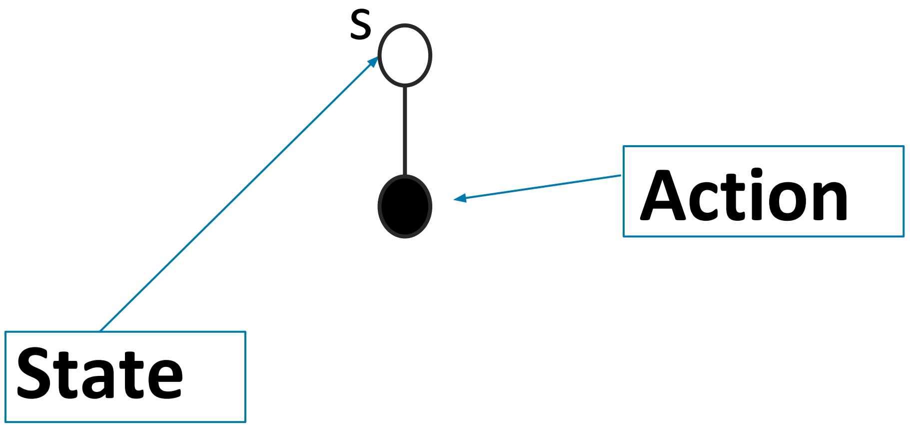
- Initialize $V_0(s) = 0$ for all s
- For i=1 until convergence*
  - For all s in S:

$$V_k^\pi(s) = R^\pi(s) + \gamma \sum_{s' \in S} P^\pi(s'|s) V_{k-1}^\pi(s')$$

- In finite horizon case, $V_k^\pi(s)$ is exact value of *k*-horizon value of state s under policy π
- In infinite horizon case, $V_k^\pi(s)$ is an estimate of infinite horizon value of state s
  - $\mathbf{V^\pi(s) = \mathbb{E}_\pi [G_t | s_t = s] \cong \mathbb{E}_\pi [r_t + \gamma V_{i-1} | s_t = s]}$

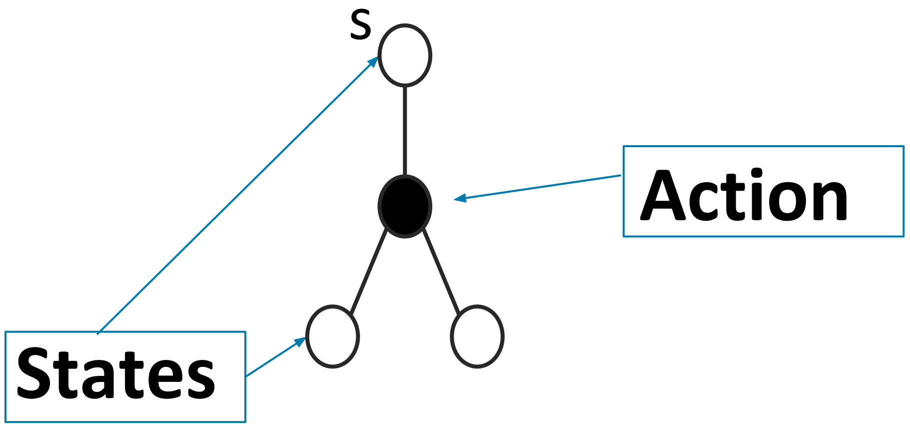
Dynamic Programming Policy Evaluation

$$V^{\pi}(s) \leftarrow \mathbb{E}_{\pi}[r_t + \gamma V_{i-1} | s_t = s]$$



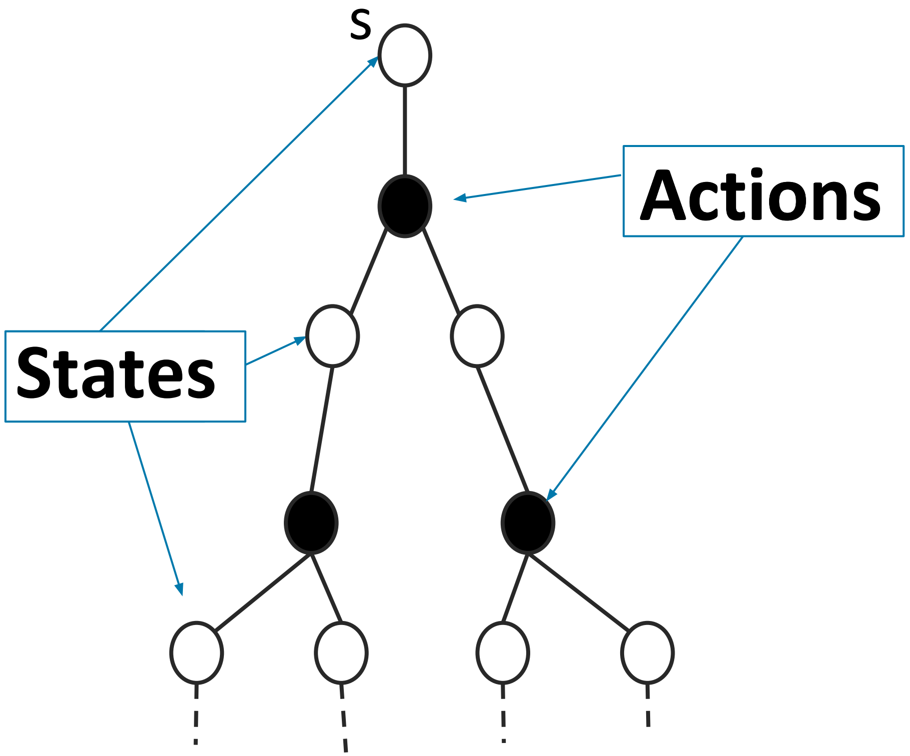
Dynamic Programming Policy Evaluation

$$V^{\pi}(s) \leftarrow \mathbb{E}_{\pi}[r_t + \gamma V_{i-1} | s_t = s]$$



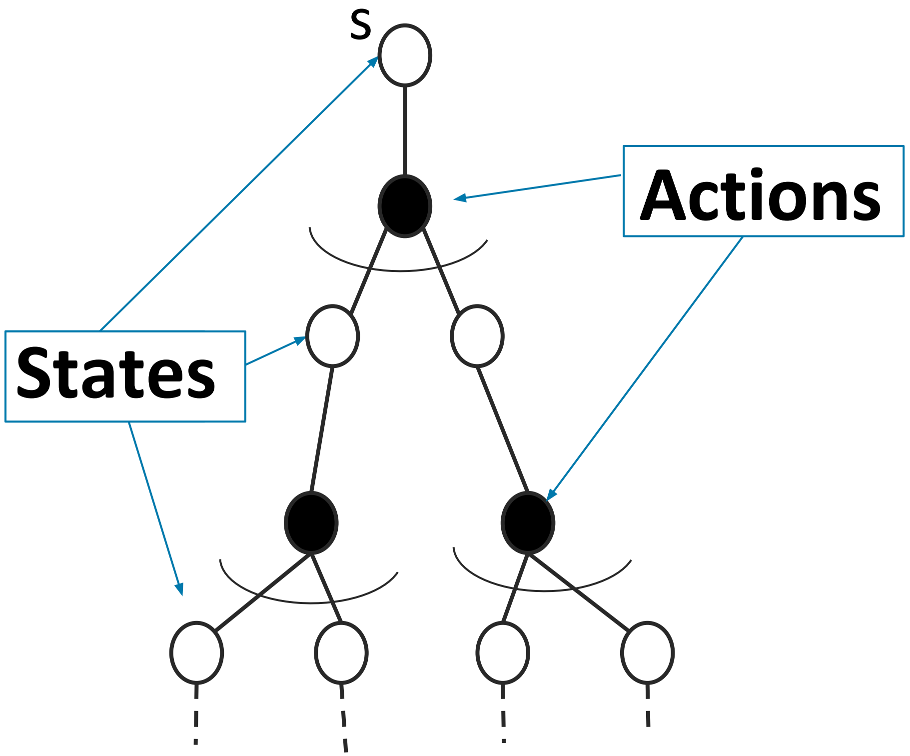
Dynamic Programming Policy Evaluation

$$V^{\pi}(s) \leftarrow \mathbb{E}_{\pi}[r_t + \gamma V_{i-1} | s_t = s]$$



Dynamic Programming Policy Evaluation

$$V^{\pi}(s) \leftarrow \mathbb{E}_{\pi}[r_t + \gamma V_{i-1} | s_t = s]$$

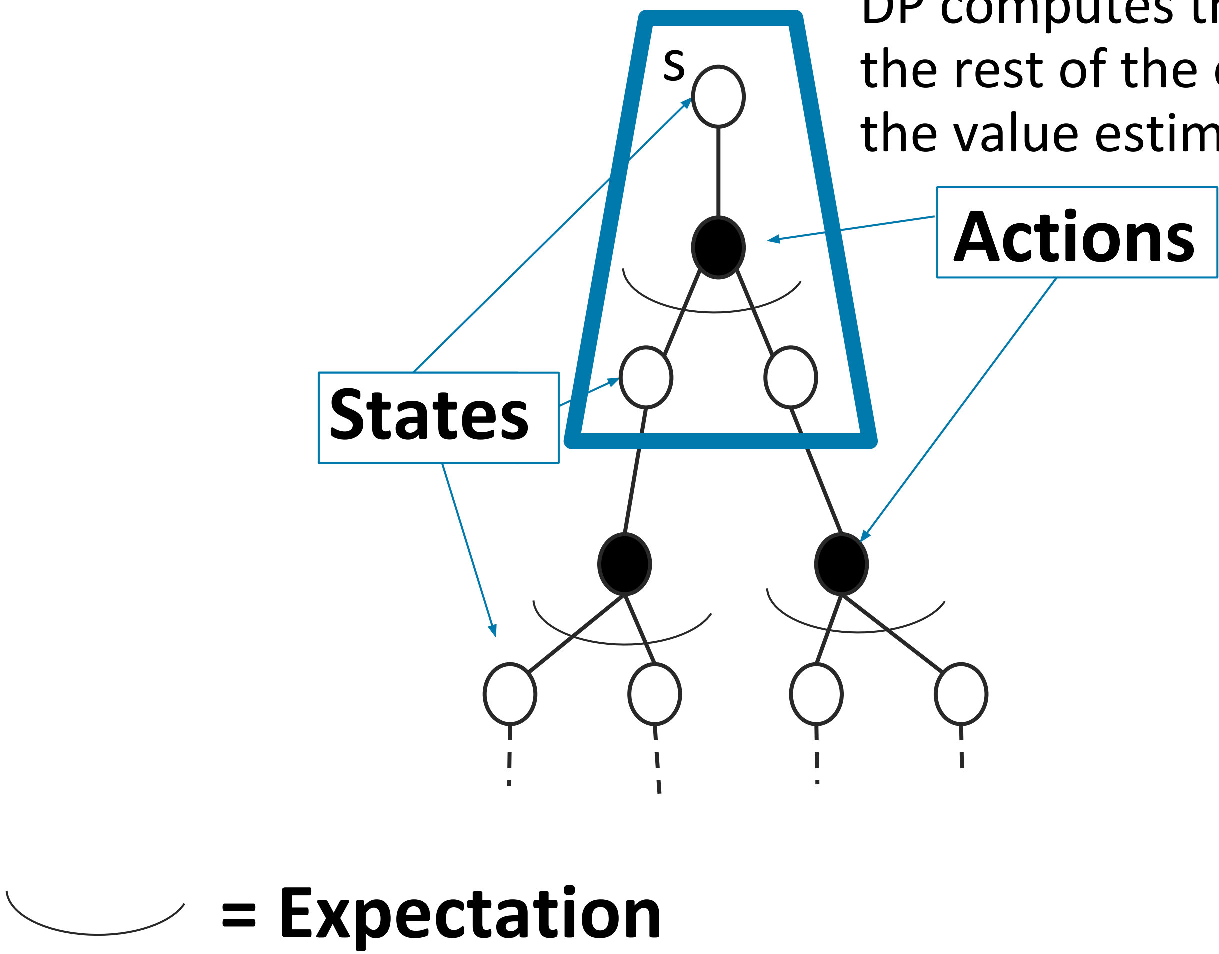


= Expectation

Dynamic Programming Policy Evaluation

$$V^{\pi}(s) \leftarrow \mathbb{E}_{\pi}[r_t + \gamma \mathbf{V}_{\mathbf{i-1}} | s_t = s]$$

DP computes this, **bootstrapping** the rest of the expected return by the value estimate V_{i-1}



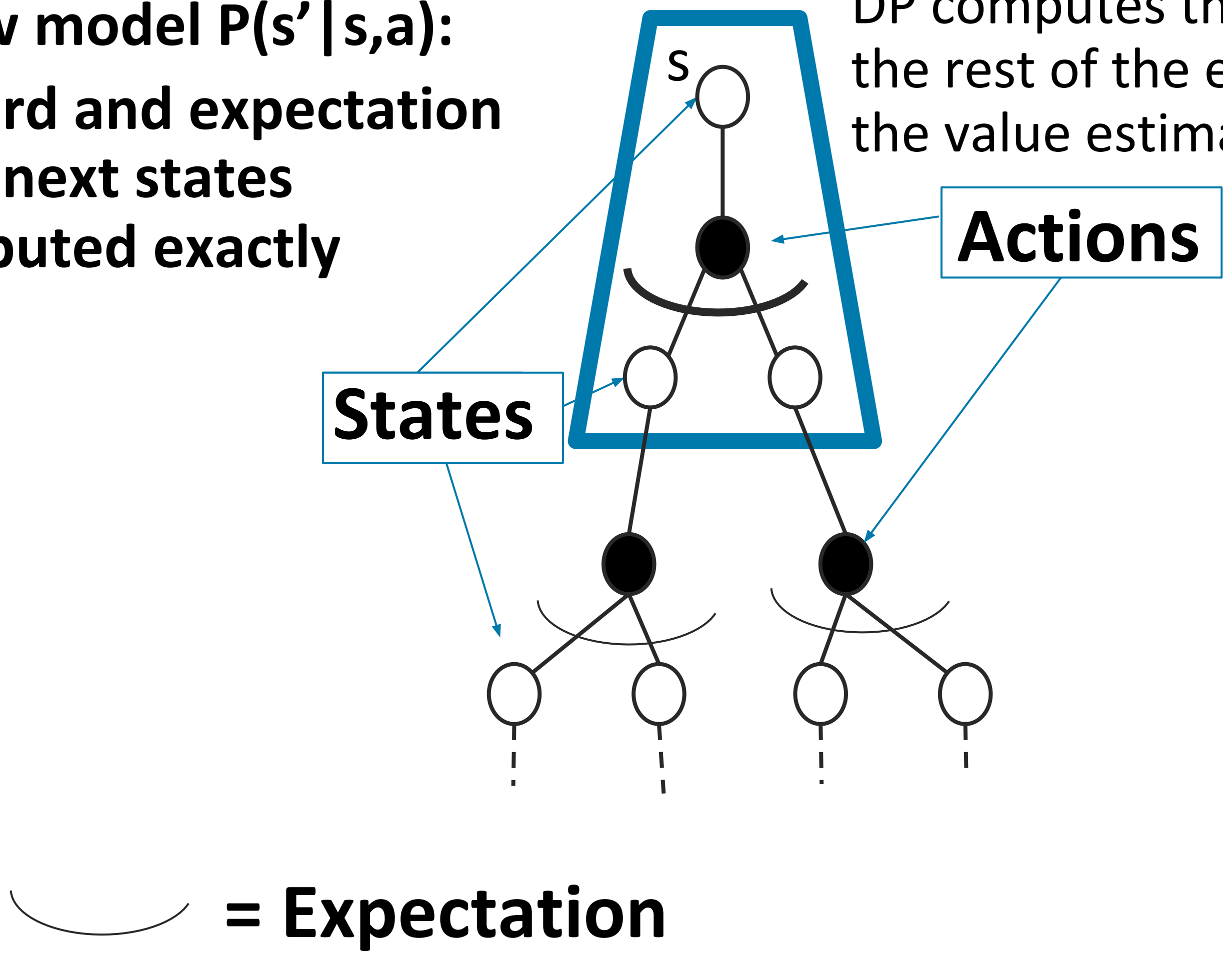
- **Bootstrapping**: Update for V uses an estimate

Dynamic Programming Policy Evaluation

$$V^{\pi}(s) \leftarrow \mathbb{E}_{\pi}[r_t + \gamma V_{i-1} | s_t = s]$$

Know model P(s'|s,a): reward and expectation over next states computed exactly

DP computes this, bootstrapping the rest of the expected return by the value estimate V_{i-1}



- Bootstrapping: Update for V uses an estimate

Policy Evaluation: $V^{\pi}(s) = \mathbb{E}_{\pi} [G_t | s_t = s]$

- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$ in MDP M under a policy π
- Dynamic programming
  - $V^{\pi}(s) \cong \mathbb{E}_{\pi}[r_t + \gamma V_{i-1} | s_t = s]$
  - Requires model of MDP M
  - Bootstraps future return using value estimate
- What if don't know how the world works?
  - Precisely, don't know dynamics model P or reward model R
- **Today: Policy evaluation without a model**
  - Given data and/or ability to interact in the environment
  - Efficiently compute a good estimate of a policy π

This Lecture: Policy Evaluation

- Dynamic programming
- **Monte Carlo policy evaluation**
  - Policy evaluation when don’t have a model of how the world work
    - Given on policy samples
    - Given off policy samples
- Temporal Difference (TD)
- Axes to evaluate and compare algorithms

Monte Carlo (MC) Policy Evaluation

- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$ in MDP M under a policy π
- $V^{\pi}(s) = \mathbb{E}_{\tau \sim \pi}[G_t | s_t = s]$
  - Expectation over trajectories τ generated by following π

Monte Carlo (MC) Policy Evaluation

- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$ in MDP M under a policy π
- $V^\pi(s) = \mathbb{E}_{\tau \sim \pi}[G_t | s_t = s]$
  - Expectation over trajectories τ generated by following π

- Simple idea: Value = mean return
- If trajectories are all finite, sample a bunch of trajectories and average returns
- By law of large numbers, average return converges to mean

Monte Carlo (MC) Policy Evaluation

- If trajectories are all finite, sample a bunch of trajectories and average returns

- Does not require MDP dynamics / rewards
- No bootstrapping
- Does not assume state is Markov
- Can only be applied to episodic MDPs
  - Averaging over returns from a complete episode
  - Requires each episode to terminate

Monte Carlo (MC) On Policy Evaluation

- Aim: estimate $V^{\pi}(s)$ given episodes generated under policy π
  - $s_1, a_1, r_1, s_2, a_2, r_2, \ldots$ where the actions are sampled from π
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$ in MDP M under a policy π
- $V^{\pi}(s) = \mathbb{E}_{\pi}[G_t | s_t = s]$
- MC computes empirical mean return
- Often do this in an incremental fashion
  - After each episode, update estimate of V^{π}

First-Visit Monte Carlo (MC) On Policy Evaluation

- After each episode $i = s_{i1}, a_{i1}, r_{i1}, s_{i2}, a_{i2}, r_{i2}, \ldots$
  - Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \ldots$ as return from time step t onwards in i-th episode
  - For each state s visited in episode i
    - For **first** time t state s is visited in episode i
      - Increment counter of total first visits $N(s) = N(s) + 1$
      - Increment total return $S(s) = S(s) + G_{i,t}$
      - Update estimate $V^{\pi}(s) = S(s) / N(s)$
- By law of large numbers, as $N(s) \rightarrow \infty$, $V^{\pi}(s) \rightarrow \mathbb{E}_{\pi} [G_t | s_t = s]$

Every-Visit Monte Carlo (MC) On Policy Evaluation

- After each episode $i = s_{i1}, a_{i1}, r_{i1}, s_{i2}, a_{i2}, r_{i2}, \ldots$
  - Define $G_{i,t}=r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \ldots$ as return from time step *t* onwards in *i*-th episode
  - For each state *s* visited in episode *i*
    - For **every** time *t* state *s* is visited in episode *i*
      – Increment counter of total visits $N(s) = N(s) + 1$
      – Increment total return $S(s) = S(s) + G_{i,t}$
      – Update estimate $V^{\pi}(s) = S(s) / N(s)$
- As $N(s) \longrightarrow \infty$, $V^{\pi}(s) \longrightarrow \mathbb{E}_{\pi} [G_t | s_t = s]$

Incremental Monte Carlo (MC) On Policy Evaluation

- After each episode i = s_{i1}, a_{i1}, r_{i1}, s_{i2}, a_{i2}, r_{i2}, …
  - Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \ldots$ as return from time step *t* onwards in *i*-th episode
  - For state *s* visited at time step *t* in episode *i*
    - Increment counter of total visits N(s) = N(s) + 1
    - Update estimate

$$V^\pi(s) = V^\pi(s)\frac{N(s)-1}{N(s)} + \frac{G_{it}}{N(s)} = V^\pi(s) + \frac{1}{N(s)}(G_{it} - V^\pi(s))$$

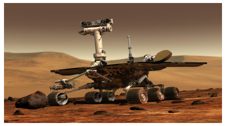
Incremental Monte Carlo (MC) On Policy Evaluation Running Mean

- After each episode *i* = s_{i1}, a_{i1}, r_{i1}, s_{i2}, a_{i2}, r_{i2}, …
  - Define $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \ldots$ as return from time step *t* onwards in *i*-th episode
  - For state *s* visited at time step *t* in episode *i*
    - Increment counter of total visits N(s) = N(s) + 1
    - Update estimate

$$V^\pi(s) = V^\pi(s) + \alpha(G_{it} - V^\pi(s))$$

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

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

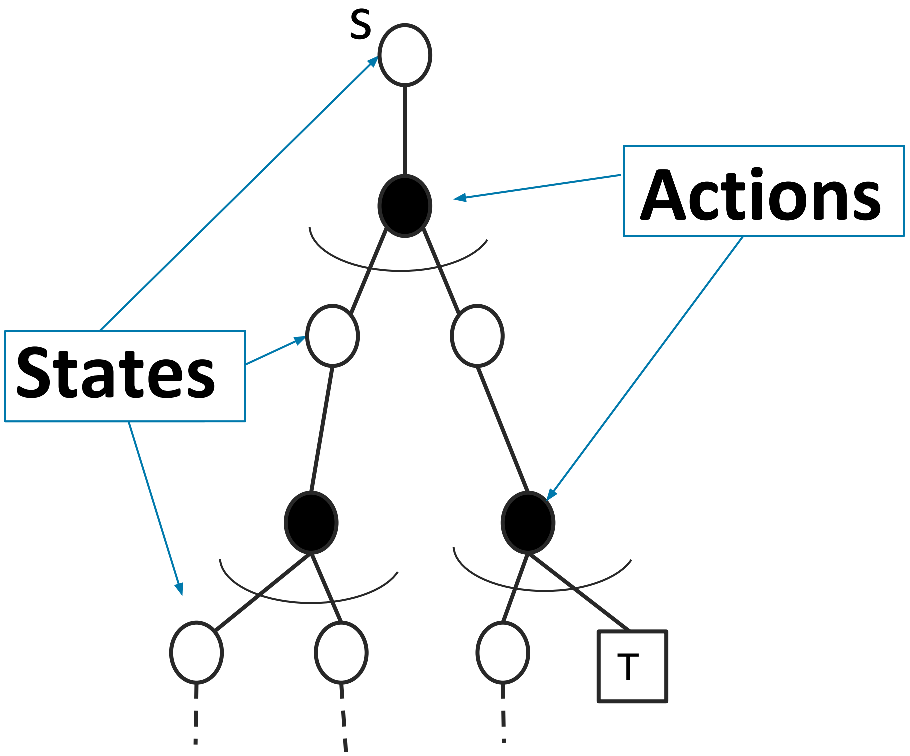
S1	S2	S3	S4	S5	S6	S7
Okay Field Site +1						Fantastic Field Site +10

- Policy: TryLeft (TL) in all states, use γ=1, S1 and S7 transition to terminal upon any action
- Start in state S3, take TryLeft, get r=0, go to S2
- Start in state S2, take TryLeft, get r=0, go to S2
- Start in state S2, take TryLeft, get r=0, go to S1
- Start in state S1, take TryLeft, get r=+1, go to terminal
- Trajectory = (S3,TL,0,S2,TL,0,S2,TL,0,S1,TL,1, terminal)
- First visit MC estimate of V of each state?

- Every visit MC estimate of S2?

MC Policy Evaluation

$$V^\pi(s) = V^\pi(s) + \alpha(G_{it} - V^\pi(s))$$



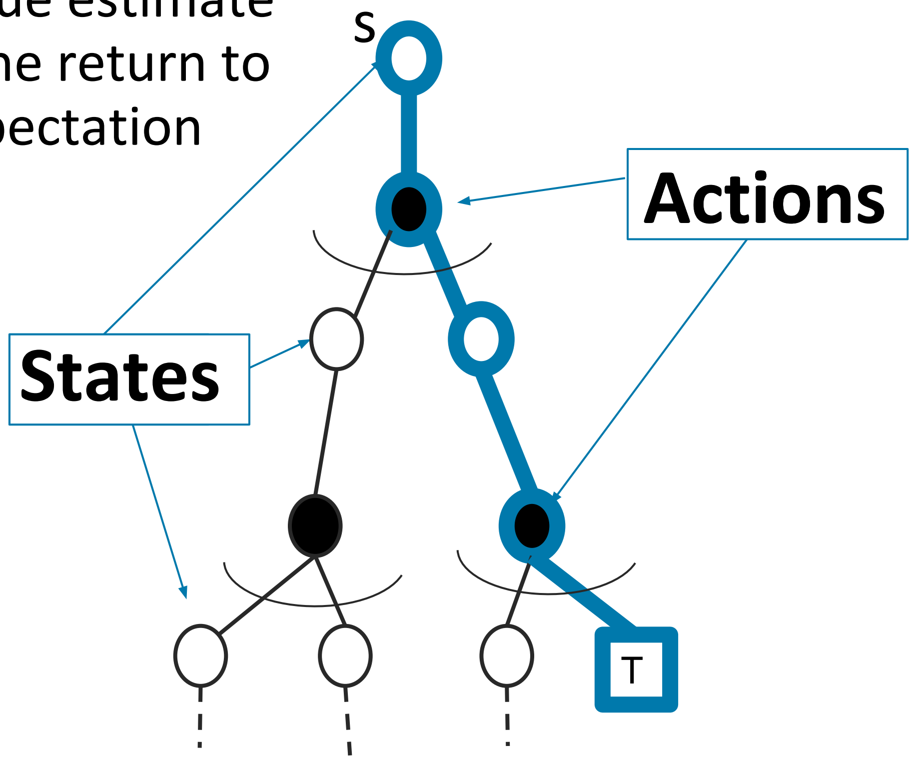
= Expectation

= Terminal state

MC Policy Evaluation

$$V^\pi(s) = V^\pi(s) + \alpha(G_{it} - V^\pi(s))$$

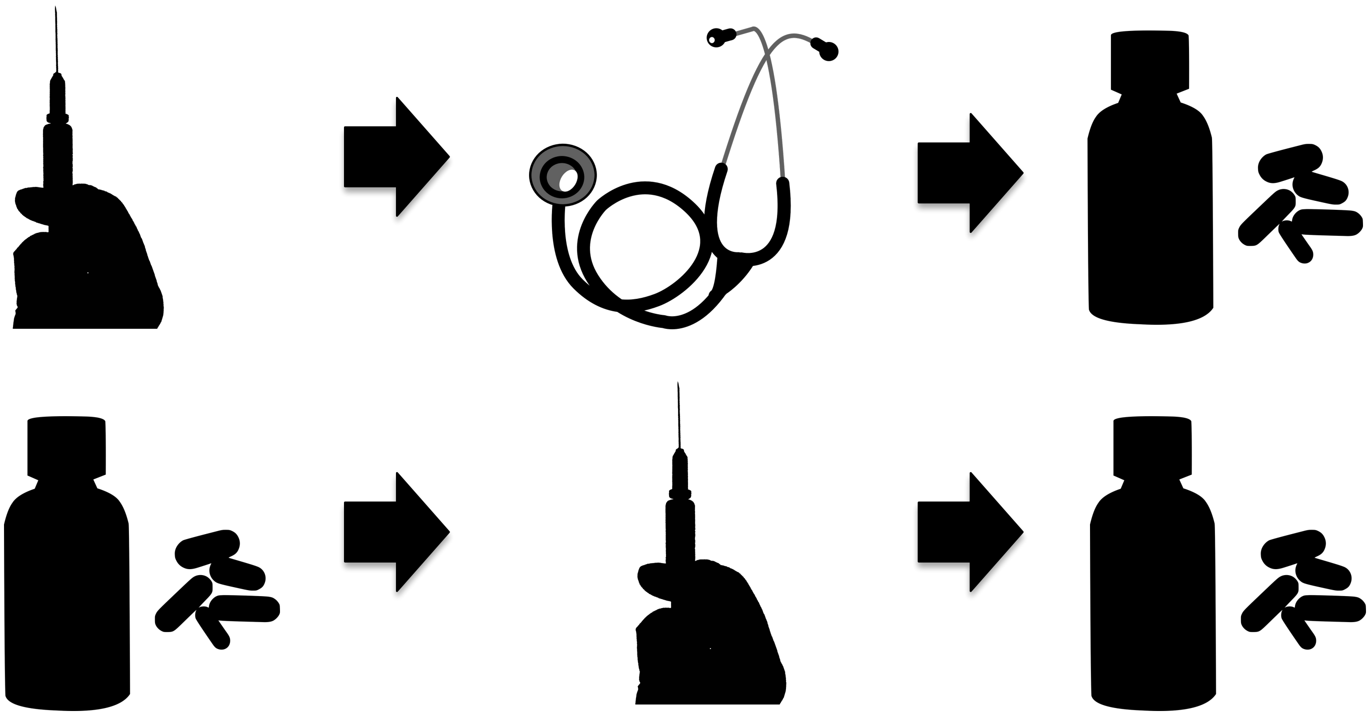
MC updates the value estimate using a **sample** of the return to approximate an expectation




= Expectation

= Terminal state

MC Off Policy Evaluation



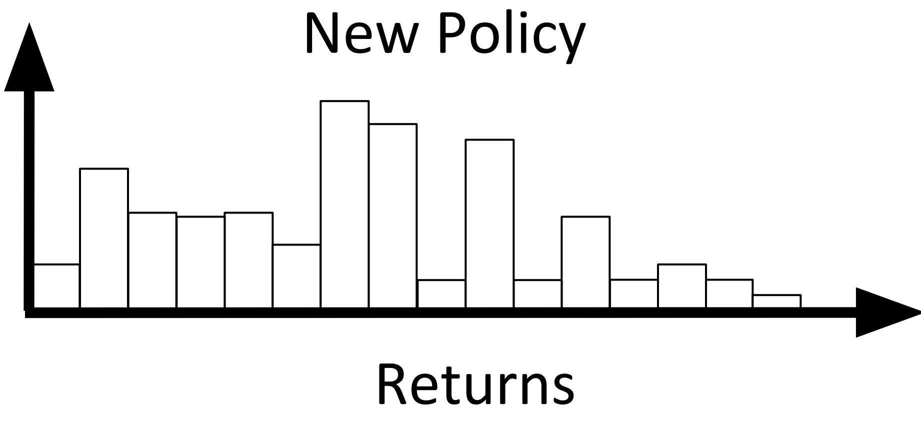
- Sometimes trying actions out is costly or high stakes
- Would like to use old data about policy decisions and their outcomes to estimate the potential value of an alternate policy

Monte Carlo (MC) **Off Policy** Evaluation

- Aim: estimate given episodes generated under policy π_1
  - $s_1, a_1, r_1, s_2, a_2, r_2, \ldots$ where the actions are sampled from π_1
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$ in MDP M under a policy π
- $V^\pi(s) = \mathbb{E}_\pi [G_t | s_t = s]$
- Have data from another policy
- If π_1 is stochastic can often use it to estimate the value of an alternate policy (formal conditions to follow)
- Again, no requirement for model nor that state is Markov

Monte Carlo (MC) **Off Policy** Evaluation: Distribution Mismatch

- Distribution of episodes & resulting returns differs between policies



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 data points is an estimate of the mean of the Gaussian.
- Definition: the bias of an estimator $\hat{\theta}$ is:

$$Bias_{\theta}(\hat{\theta}) = \mathbb{E}_{x|\theta}[\hat{\theta}] - \theta \tag{1}$$

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

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

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

$$MSE(\hat{\theta}) = Var(\hat{\theta}) + Bias_{\theta}(\hat{\theta})^2 \tag{3}$$

Importance Sampling

- Goal: estimate the expected value of a function $f(x)$ under some probability distribution $p(x)$, $\mathbb{E}_{x \sim p}[f(x)]$
- Have data $x_1, x_2, \ldots, x_n$ sampled from distribution $q(s)$
- Under a few assumptions, can use samples to obtain an unbiased estimate of $\mathbb{E}_{x \sim q}[f(x)]$

$$\mathbb{E}_{x \sim q}[f(x)] = \int_x q(x) f(x)$$

Importance Sampling for Policy Evaluation

- Aim: estimate V^{π_1} given episodes generated under policy π_2
  - $s_1, a_1, r_1, s_2, a_2, r_2, \ldots$ where the actions are sampled from π_2
- Have access to $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$ in MDP M under a policy π_2
- Want $V^{\pi_1}(s) = E_{\pi_1}[G_t | s_t = s]$
- Have data from another policy
- If π_2 is stochastic can often use it to estimate the value of an alternate policy (formal conditions to follow)
- Again, no requirement for model nor that state is Markov

Importance Sampling (IS) for Policy Evaluation

- Let h be a particular episode (history) of states, actions and rewards

$$h = (s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_{terminal})$$

Probability of a Particular Episode

- Let h be a particular episode (history) of states, actions and rewards

$$h = (s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_{terminal})$$

$$\begin{aligned} p(h_j|\pi, s) &= p(a_{j1}|s_{j1})p(r_{j1}|s_{j1}, a_{j1})p(s_{j2}|s_{j1}, a_{j1})p(a_{j2}|s_{j2})\ldots \\ &= \prod_{t=1}^{L_j-1} p(a_{j,t}|s_{j,t})p(r_{j,t}|s_{j,t}, a_{j,t})p(s_{j,t+1}|s_{j,t}, a_{j,t}) \\ &= \prod_{t=1}^{L_j-1} \pi(a_{j,t}|s_{j,t})p(r_{j,t}|s_{j,t}, a_{j,t})p(s_{j,t+1}|s_{j,t}, a_{j,t}) \end{aligned}$$

Importance Sampling (IS) for Policy Evaluation

- Let h be a particular episode (history) of states, actions and rewards

$$h = (s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_{terminal})$$

$$V^{\pi_1}(s) \approx \frac{1}{N} \sum_{j=1}^{N} \frac{p(h_j|\pi_1, s)}{p(h_j|\pi_2, s)} G(h_j)$$

Importance Sampling for Policy Evaluation

- Aim: estimate V^{π_1} given episodes generated under policy π_2
  - $s_1, a_1, r_1, s_2, a_2, r_2, \ldots$ where the actions are sampled from π_2
- Have access to $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$ in MDP M under a policy π_2
- Want $V^{\pi_1}(s) = E_{\pi_1}[G_t | s_t = s]$
- IS = Monte Carlo estimate given off policy data
- Model-free method
- Does not require Markov assumption
- Under some assumptions, unbiased & consistent estimator of V^{π_1}
- Can be used when agent is interacting with environment to estimate value of policies different than agent's control policy
- More later this quarter about batch learning

Monte Carlo (MC) Policy Evaluation Summary

- Aim: estimate $V^{\pi}(s)$ given episodes generated under policy π
  - $s_1, a_1, r_1, s_2, a_2, r_2, \ldots$ where the actions are sampled from π
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$ in MDP M under a policy π
- $V^{\pi}(s) = \mathbb{E}_{\pi}[G_t | s_t = s]$
- Simple: Estimates expectation by empirical average (given episodes sampled from policy of interest) or reweighted empirical average (importance sampling)
- Updates value estimate by using a **sample** of return to approximate the expectation
- No bootstrapping
- Converges to true value under some (generally mild) assumptions

Monte Carlo (MC) Policy Evaluation Key Limitations

- Generally high variance estimator
  - Reducing variance can require a lot of data
- Requires episodic settings
  - Episode must end before data from that episode can be used to update the value function

This Lecture: Policy Evaluation

- Dynamic programming
- Monte Carlo policy evaluation
  - Policy evaluation when don’t have a model of how the world work
    - Given on policy samples
    - Given off policy samples
- **Temporal Difference (TD)**
- Axes to evaluate and compare algorithms

Temporal Difference Learning

- "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*
- Combination of Monte Carlo & dynamic programming methods
- Model-free
- **Bootstraps and samples**
- Can be used in episodic or infinite-horizon non-episodic settings
  - Immediately updates estimate of V after each (s,a,r,s') tuple

Temporal Difference Learning for Estimating V

- Aim: estimate $V^\pi(s)$ given episodes generated under policy π
- $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$ in MDP M under a policy π
- $V^\pi(s) = \mathbb{E}_\pi [G_t | s_t = s]$
- Recall Bellman operator (if know MDP models)

$$B^\pi V(s) = R^\pi(s) + \gamma \sum_{s' \in S} P^\pi(s'|s) V(s')$$

- In incremental every-visit MC, update estimate using 1 sample of return (for the current i^{th} episode)

$$V^\pi(s_{it}) = V^\pi(s_{it}) + \alpha(G_{it} - V^\pi(s_{it}))$$

- Insight: have an estimate of V^π, use to estimate expected return

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

Temporal Difference [TD(0)] Learning

- Aim: estimate $V^{\pi}(s)$ given episodes generated under policy π
  - s_1, a_1, r_1, s_2, a_2, r_2, … where the actions are sampled from π

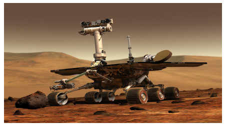
- Simplest TD learning: update value towards estimated value

$$V^{\pi}(s_t) = V^{\pi}(s_t) + \alpha\left(\underbrace{[r_t + \gamma V^{\pi}(s_{t+1})]}_{\text{TD target}} - V^{\pi}(s_t)\right)$$

- TD error:

$$\delta_t = r_t + \gamma V^{\pi}(s_{t+1}) - V^{\pi}(s_t))$$

- **Can immediately update value estimate after (s,a,r,s') tuple**
- **Don't need episodic setting**

S1	S2	S3	S4	S5	S6	S7
Okay Field Site +1						Fantastic Field Site +10

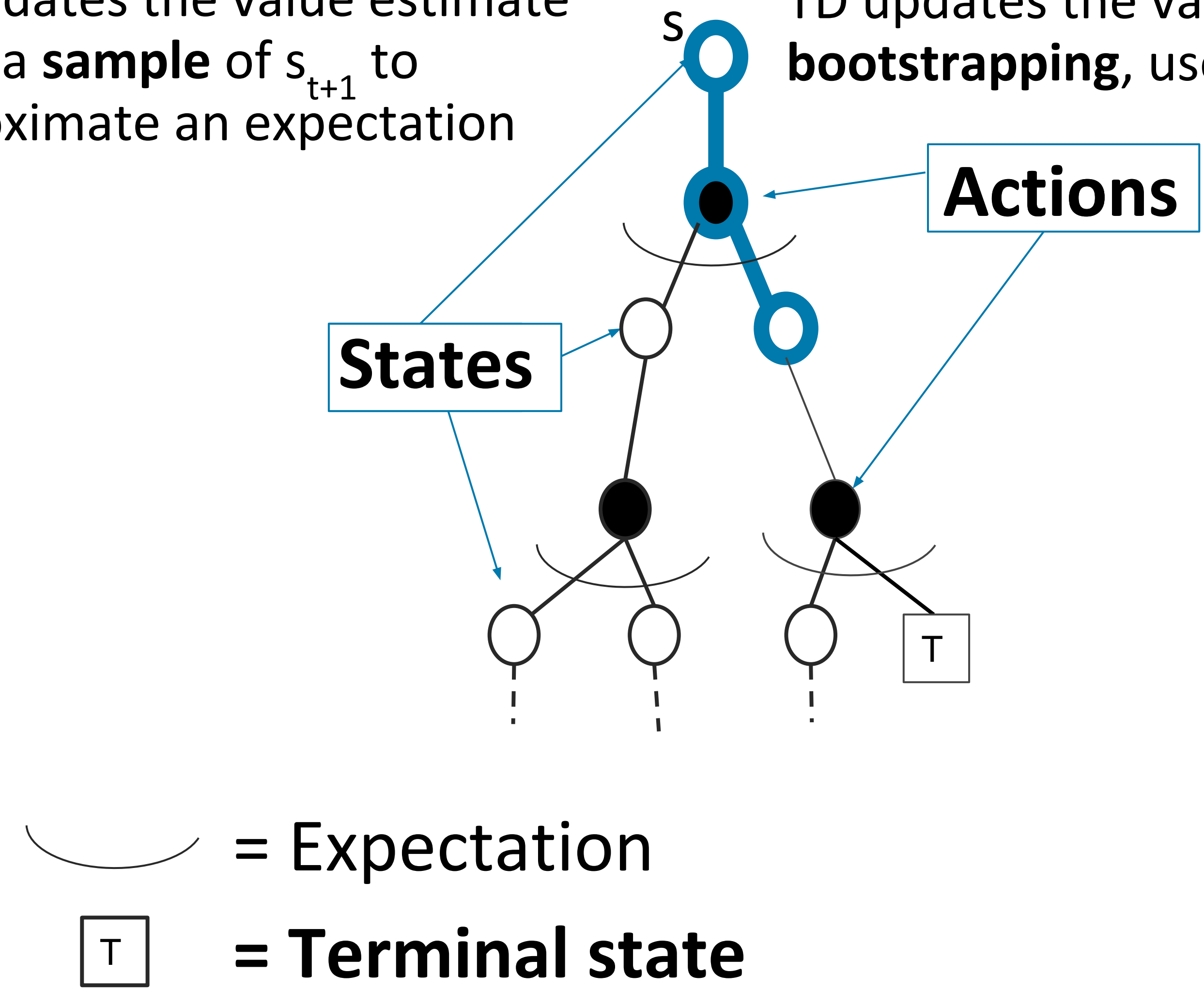
- Policy: TryLeft (TL) in all states, use γ=1, S1 and S7 transition to terminal upon any action
- Start in state S3, take TryLeft, get r=0, go to S2
- Start in state S2, take TryLeft, get r=0, go to S2
- Start in state S2, take TryLeft, get r=0, go to S1
- Start in state S1, take TryLeft, get r=+1, go to terminal
- Trajectory = (S3,TL,0,S2,TL,0,S2,TL,0,S1,TL,1, terminal)
- First visit MC estimate of all states? [1 1 1 0 0 0 0]
- Every visit MC estimate of S2? 1
- TD estimate of all states (init at 0) with alpha = 1?

Temporal Difference Policy Evaluation

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

TD updates the value estimate using a **sample** of s_{t+1} to approximate an expectation

TD updates the value estimate by **bootstrapping**, uses estimate of $V(s_{t+1})$



This Lecture: Policy Evaluation

- Dynamic programming
- Monte Carlo policy evaluation
  - Policy evaluation when don’t have a model of how the world work
    - Given on policy samples
    - Given off policy samples
- Temporal Difference (TD)
- **Axes to evaluate and compare algorithms**

Some Important Properties to Evaluate Policy Evaluation Algorithms

- Usable when no models of current domain
  - DP: No MC: Yes TD: Yes
- Handles continuing (non-episodic) domains
  - DP: Yes MC: No TD: Yes
- Handles Non-Markovian domains
  - DP: No MC: Yes TD: No
- Converges to true value in limit*
  - DP: Yes MC: Yes TD: Yes
- Unbiased estimate of value
  - DP: NA MC: Yes TD: No

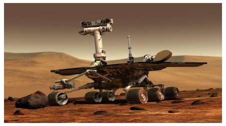
* For tabular representations of value function. More on this in later lectures

Some Important Properties to Evaluate Model-free Policy Evaluation Algorithms

- Bias/variance characteristics
- Data efficiency
- Computational efficiency

Bias/Variance of Model-free Policy Evaluation Algorithms

- Return G_t is an unbiased estimate of $V^\pi(s_t)$
- TD target $[r_t + \gamma V^\pi(s_{t+1})]$ is a biased estimate of $V^\pi(s_t)$
- But often much lower variance than a single return G_t
- Return function of multi-step seq. of random actions, states & rewards
- TD target only has one random action, reward and next state
- MC
  - Unbiased
  - High variance
  - Consistent (converges to true) even with function approximation
- TD
  - Some bias
  - Lower variance
  - TD(0) converges to true value with tabular representation
  - TD(0) does not always converge with function approximation

S1	S2	S3	S4	S5	S6	S7
Okay Field Site +1						Fantastic Field Site +10

- Policy: TryLeft (TL) in all states, use γ=1, S1 and S7 transition to terminal upon any action
- Start in state S3, take TryLeft, get r=0, go to S2
- Start in state S2, take TryLeft, get r=0, go to S2
- Start in state S2, take TryLeft, get r=0, go to S1
- Start in state S1, take TryLeft, get r=+1, go to terminal
- Trajectory = (S3,TL,0,S2,TL,0,S2,TL,0,S1,TL,1, terminal)
- **Recall**
- First visit MC estimate of all states? [1 1 1 0 0 0 0]
- Every visit MC estimate of S2? 1
- TD estimate of all states (init at 0) [1 0 0 0 0 0 0] with alpha = 1
- TD(0) only uses a data point (s,a,r,s') once
- Monte Carlo takes entire return from s to end of episode

Batch MC and TD

- Batch (Offline) solution for finite dataset
  - Given set of K episodes
  - Repeatedly sample an episode from K
  - Apply MC or TD(0) to that episode
- What do MC and TD(0) converge to?

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

- Two states A, B; γ=1; 8 episodes of experience

A, 0, B, 0	B, 1
B, 1	B, 1
B, 1	B, 1
B, 1	B, 0

What is V(A), V(B)?

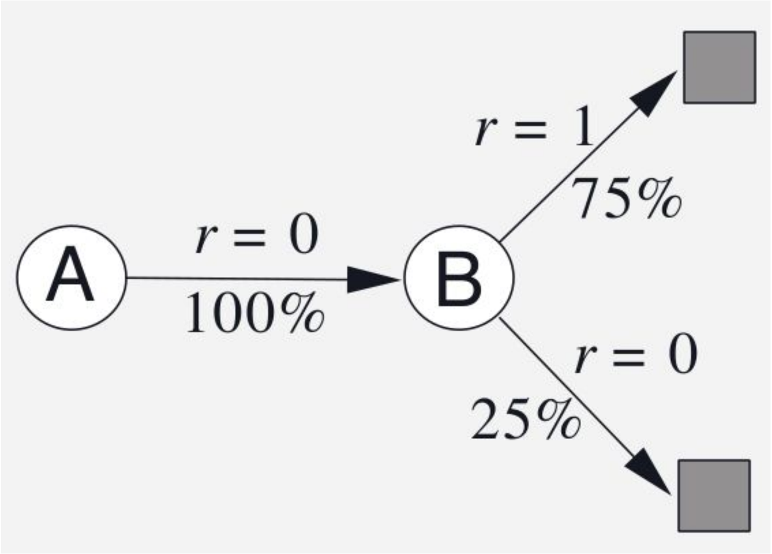
AB Example: (Ex.6.4, Sutton & Barto, 2018)

- Two states A, B; γ=1; 8 episodes of experience

A, 0, B, 0	B, 1
B, 1	B, 1
B, 1	B, 1
B, 1	B, 0

What is V(A), V(B)?

- V(B) = .75 (by TD or MC)
- V(A)?



Batch MC and TD: Converges

- Monte Carlo in batch setting converges to min MSE (mean squared error)
  - Minimize loss with respect to observed returns
  - In AB example, V(A) = 0
- TD(0) converges to DP policy V^π for the MDP with the maximum likelihood model estimates
  - Maximum likelihood Markov decision process model

$$\hat{P}(s'|s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k-1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a, s_{k,t+1} = s')$$

$$\hat{r}(s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k-1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a) r_{t,k}$$

  - Compute V^π using this model
  - In AB example, V(A) = 0.75

Some Important Properties to Evaluate Model-free Policy Evaluation Algorithms

- Data efficiency & Computational efficiency
- In simplest TD, use (s,a,r,s') once to update V(s)
  - O(1) operation per update
  - In an episode of length L, O(L)
- In MC have to wait till episode finishes, then also O(L)
- MC can be more data efficient than simple TD
- But TD exploits Markov structure
  - If in Markov domain, leveraging this is helpful

Alternative: Certainty Equivalence V^π MLE MDP Model Estimates

- Model-based option for policy evaluation without true models
- After each (s,a,r,s') tuple
  - Recompute maximum likelihood MDP model for (s,a)

$$\hat{P}(s'|s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k-1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a, s_{k,t+1} = s')$$

$$\hat{r}(s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k-1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a) r_{t,k}$$

  - Compute V^π using MLE MDP* (e.g. see method from lecture 2)
- *Requires initializing for all (s,a) pairs

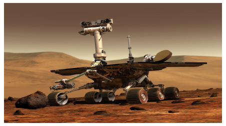
Alternative: Certainty Equivalence V^π MLE MDP Model Estimates

- Model-based option for policy evaluation without true models
- After each (s,a,r,s') tuple
  - Recompute maximum likelihood MDP model for (s,a)

$$\hat{P}(s'|s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k - 1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a, s_{k,t+1} = s')$$

$$\hat{r}(s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k - 1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a) r_{t,k}$$

  - Compute V^π using MLE MDP* (e.g. see method from lecture 2)
- *Requires initializing for all (s,a) pairs
- Cost: Updating MLE model and MDP planning at each update ($O(|S|^3$ for analytic matrix soln, $O(|S|^2|A|)$ for iterative methods)
- Very data efficient and very computationally expensive
- Consistent
- Can also easily be used for off-policy evaluation

S1	S2	S3	S4	S5	S6	S7
Okay Field Site +1						Fantastic Field Site +10

- Policy: TryLeft (TL) in all states, use γ=1, S1 and S7 transition to terminal upon any action
- Start in state S3, take TryLeft, get r=0, go to S2
- Start in state S2, take TryLeft, get r=0, go to S2
- Start in state S2, take TryLeft, get r=0, go to S1
- Start in state S1, take TryLeft, get r=+1, go to terminal
- Trajectory = (S3,TL,0,S2,TL,0,S2,TL,0,S1,TL,1, terminal)
- **Recall**
- First visit MC estimate of all states? [1 1 1 0 0 0 0]
- Every visit MC estimate of S2? 1
- TD estimate of all states (init at 0) [1 0 0 0 0 0 0] with alpha = 1
- TD(0) only uses a data point (s,a,r,s') once
- Monte Carlo takes entire return from s to end of episode
- What is certainty equivalent estimate?

Some Important Properties to Evaluate Policy Evaluation Algorithms

- Robustness to Markov assumption
- Bias/variance characteristics
- Data efficiency
- Computational efficiency

Summary: Policy Evaluation

- Dynamic programming
- Monte Carlo policy evaluation
  - Policy evaluation when don’t have a model of how the world work
    - Given on policy samples
    - Given off policy samples
- Temporal Difference (TD)
- Axes to evaluate and compare algorithms

Class Structure

- Last Time:
  - Markov reward / decision processes
  - Policy evaluation & control when have true model (of how the world works)
- Today:
  - Policy evaluation when don't have a model of how the world works
- **Next time:**
  - **Control when don't have a model of how the world works**