Approximate Methods : Model Free Prediction

Easwar Subramanian

TCS Innovation Labs, Hyderabad

Email : easwar.subramanian@tcs.com / cs5500.2020@iith.ac.in

September 2, 2022

Overview

1. Review
2. DP Algorithms : A Closer Look
3. Model Free Prediction: Monte Carlo Methods
4. Model Free Prediction : Temporal Difference

Review

MDP and RL setting

- **MDP Setting** : The agent has knowledge of the state transition matrices $\mathcal{P}^a_{ss'}$ and the reward function $\mathcal{R}$
- **RL Setting** : The agent <u>does not</u> have knowledge of the state transition matrices $\mathcal{P}^a_{ss'}$ and the reward function $\mathcal{R}$

Course Setup

Markov Decision Process and Dynamic Programming

↓

Modern Reinforcement Learning

↓

Deep Reinforcement Learning

DP Algorithms : A Closer Look

DP Algorithms : Terminology

Let us consider the policy evaluation formula

$$V_{k+1}(s) \leftarrow \sum_{a} \pi(s,a) \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V_k(s')\right]$$

DP Algorithms : Terminology

Let us consider the policy evaluation formula

One-step lookahead

$$V_{k+1}(s) \leftarrow \sum_{a} \pi(s, a) \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V_k(s')\right]$$

DP Algorithms : Terminology

Let us consider the policy evaluation formula

One-step lookahead

$$V_{k+1}(s) \leftarrow \sum_{a} \pi(s,a) \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V_k(s') \right]$$

Backup

DP Algorithms : Terminology

Let us consider the policy evaluation formula

One-step lookahead

$$V_{k+1}(s) \leftarrow \sum_{a} \pi(s, a) \boxed{\sum_{s'}} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V_k(s') \right]$$

Full backup

Backup

DP Algorithms : Terminology

Let us consider the policy evaluation formula

Bootstrap

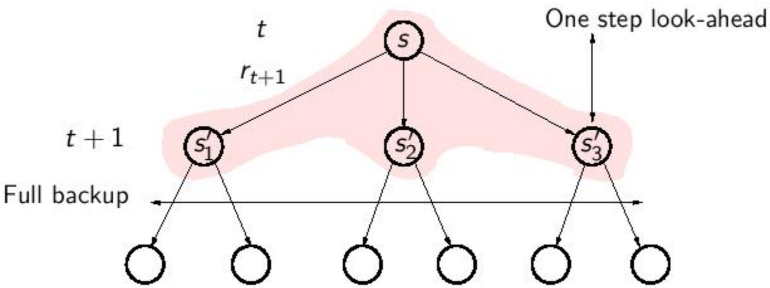
One-step lookahead

$$V_{k+1}(s) \leftarrow \sum_a \pi(s,a) \boxed{\sum_{s'}} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V_k(s')\right]$$

Full backup

Backup

DP Algorithms : Schematic View



$$V^{\pi}(s) = \sum_{a} \pi(a|s) \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V^{\pi}(s') \right]$$

$$V_{k+1}(s) \leftarrow \sum_{a} \pi(a|s) \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V_{k}(s') \right]$$

Drawbacks of DP Algorithms

- Requires full prior knowledge of the dynamics of the environment
- Can be implemented only on small or medium sized discrete state spaces
  - ★ For large problems, DP suffers from Bellman's curse of dimensionality
- DP uses full width back-ups
  - ★ Every successor state and action is considered

Model Free Prediction: Monte Carlo Methods

Model Free Prediction : Key Idea

$$
\begin{aligned}
V^{\pi}(s) &\stackrel{\text{def}}{=} \mathbb{E}_{\pi}(G_t | s_t = s) = \mathbb{E}_{\pi}\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right) \\
&= \mathbb{E}_{\pi}\left[r_{t+1} + \gamma V^{\pi}(s_{t+1}) | s_t = s\right]
\end{aligned}
$$

How can we estimate the expectations?
Use samples!

Monte Carlo Policy Evaluation

- Goal : Evaluate $V^\pi(s)$ using experiences (or trajectories) under policy π

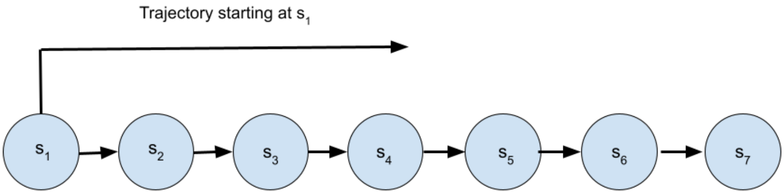
$$s_0, a_0, r_1, s_1, a_1, r_2, s_3, \cdots, s_T$$

- Recall that

$$V^\pi(s) = \mathbb{E}_\pi(G_t | s_t = s) = \mathbb{E}_\pi \left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \right)$$

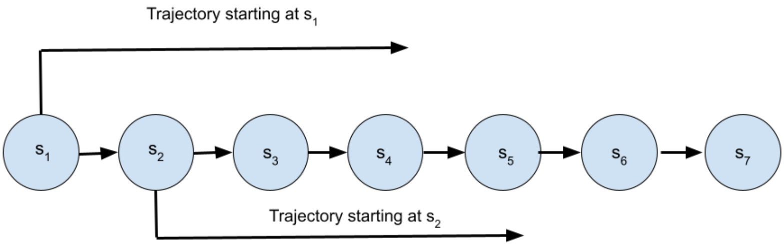
- The idea is to calculate **sample** mean return (G_t) starting from state s instead of expected mean return

Monte Carlo Evalution : Schematics



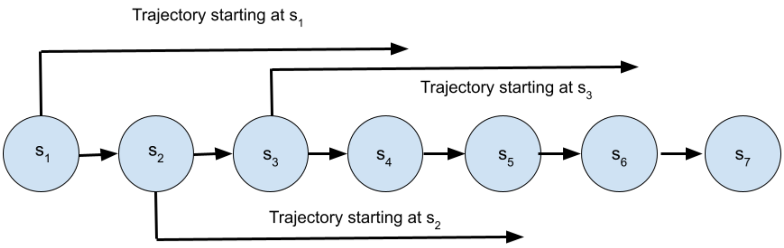
- Use G_1 to update $V^{\pi}(s_1)$

Monte Carlo Evalution : Schematics



- Use G_1 to update $V^\pi(s_1)$
- Use G_2 to update $V^\pi(s_2)$

Monte Carlo Evalution : Schematics



- Use G_1 to update $V^\pi(s_1)$
- Use G_2 to update $V^\pi(s_2)$
- Use G_3 to update $V^\pi(s_3)$

First-visit Monte Carlo Policy Evaluation

- To evaluate $V^{\pi}(s)$ for some given state s, repeat over several episodes
  - ★ The first time t that $s_t = s$ in the episode
    1. Increment counter for number of visits to s: $N(s) \leftarrow N(s) + 1$
    2. Increment running sum of total returns with return from current episode: $S(s) \leftarrow S(s) + G_t$
- Monte Carlo estimate of value function $V(s) \leftarrow S(s)/N(s)$

By the law of large numbers $V(s) \rightarrow V^{\pi}(s)$ as number of episodes increases

Every-visit Monte Carlo Policy Evaluation

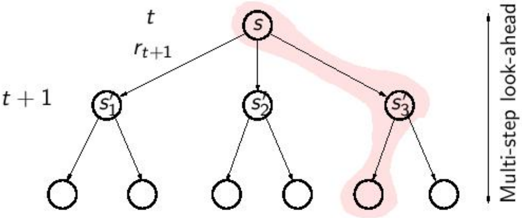
- To evaluate $V^{\pi}(s)$ for some given state s, repeat over several episodes
  - ★ Every time t that $s_t = s$ in the episode
    1. Increment counter for number of visits to s: $N(s) \leftarrow N(s) + 1$
    2. Increment running sum of total returns with return from current episode: $S(s) \leftarrow S(s) + G_t$
- Monte Carlo estimate of value function $V(s) \leftarrow S(s)/N(s)$

By the law of large numbers $V(s) \to V^{\pi}(s)$ as number of episodes increases

Monte Carlo Method : Example

- Consider an MDP with two states $\mathcal{S} = \{A, B\}$ with $\gamma = 1$
- $\mathcal{P}$ and $\mathcal{R}$ are unknown
- Consider a policy π that gives rise to following state-reward sequence
  - ★ $A(+3), A(+2), B(-4), A(+4), B(-3)$
  - ★ $B(-2), A(+3), B(-3)$
- What is $V^{\pi}(A)$ and $V^{\pi}(B)$ if we use first visit MC and every visit MC respectively ?
- First visit MC : $V(A) = \frac{1}{2}(2+0) = 1$; $V(B) = \frac{1}{2}(-3-2) = -5/2$
- Every visit MC : $V(A) = \frac{1}{4}(2-1+1+0) = 1/2$; $V(B) = \frac{1}{4}(-3-3-3-2) = -11/4$

Monte Carlo Algorithms: A Schematic View



- Uses experience, rather than model
- Uses only experience; does not bootstrap
- Needs complete sequences; suitable only for episodic tasks
- Suited for off-line learning
- Time required for one estimate does not depend on total number of states
- Estimates for each state are independent

Model Free Prediction : Temporal Difference

Temporal Difference : Key Idea

$$
\begin{aligned}
V^\pi(s) &\stackrel{\text{def}}{=} \mathbb{E}_\pi(G_t|s_t = s) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}|s_t = s\right) \\
&= \mathbb{E}_\pi\left[r_{t+1} + \gamma V^\pi(s_{t+1})|s_t = s\right]
\end{aligned}
$$

- Estimate expectation from experience using the recursive decomposition formulation of the value function

Incremental Calculation of Mean

$$
\begin{aligned}
\mu_{k+1} &\stackrel{\text{def}}{=} \frac{1}{k+1}\sum_{i=1}^{k+1} x_i \\
&= \frac{1}{k+1}\sum_{i=1}^{k} x_i + \frac{1}{k+1}x_{k+1} \\
&= \frac{k}{k+1}\left(\frac{1}{k}\sum_{i=1}^{k} x_i\right) + \frac{1}{k+1}x_{k+1} \\
&= \frac{k}{k+1}\mu_k + \frac{1}{k+1}x_{k+1} \\
&= \mu_k + \frac{1}{k+1}(x_{k+1} - \mu_k)
\end{aligned}
$$

$$
\text{Update} = \text{learning rate} \times (\text{Target} - \text{Previous Value})
$$

General Form of Update Rule

The general form for the update rule that is present in the incremental calculation is,

$$\text{New Estimate} \leftarrow \text{Old Estimate} + \text{Learning Rate}(\text{Target - Old Estimate})$$

- The expression (Target - Old Estimate) is an error of the estimate
- The error is reduced by taking steps towards the "Target"
- The target is persumed to indicate a desirable direction to move
- In the incremental calculation of mean, the term x_{k+1} is the target

One-Step TD

- We wish to approximate

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[r_{t+1} + \gamma V^{\pi}(s_{t+1}) | s_t = s\right]$$

- Approximate the expectation by a sample mean
  - If the *transition* (s_t, r_{t+1}, s_{t+1}) is observed at time t under π, then

$$V(s_t) \leftarrow V(s_t) \; + \; \alpha_t[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$

  - Samples come from different visits to the state s, either from same or different trajectories
  - Compute the sample mean incrementally

One-Step TD : TD(0) Algorithm

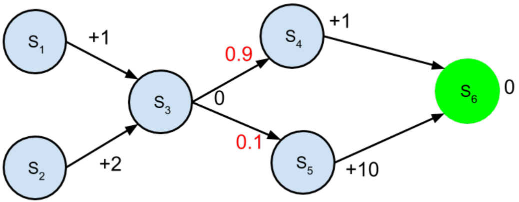
Algorithm TD(0) : Algorithm

```
Initialize V(s) arbitrarily (say, V(s) = 0  ∀s ∈ S);
for k = 1, 2, ··· , K do
    Let s be a start state for episode k
    for For each step in the k-th episode do
        Take action a recommended by policy π from state s
        Collect reward r and reach next state s′
        Perform the following TD update
```

$$V(s) = V(s) + \alpha[r + \gamma V(s') - V(s)]$$

```
        Assign s ← s′
    end for
end for
```

TD vs MC : Example



(1) $s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6$

(2) $s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_5 \xrightarrow{10} s_6$

(3) $s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6$

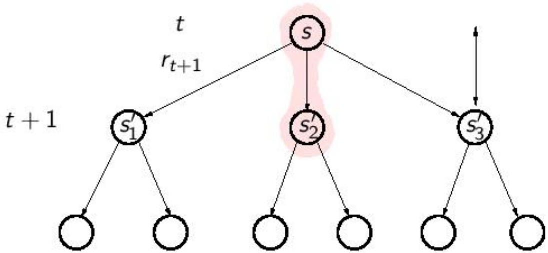
(4) $s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6$

(5) $s_2 \xrightarrow{2} s_3 \xrightarrow{0} s_5 \xrightarrow{10} s_6$

TD vs MC : Example

- True value of each state is given by
  $V(s_6) = 0, V(s_5) = 10, V(s_4) = 1, V(s_3) = 1.9, V(s_2) = 3.9$ and $V(s_1) = 2.9$
- Evaluate $V(s_1)$ and $V(s_2)$ using MC $V(s_1) = 4.25$ and $V(s_2) = 12$
- Evaluate $V(s_1)$ and $V(s_2)$ using TD
  - ★ First trajectory $(s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6)$
    $V(s_1) = 1;\ V(s_3) = 0;\ V(s_4) = 1;\ V(s_6) = 0$
  - ★ Second trajectory $(s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_5 \xrightarrow{10} s_6)$
    $V(s_1) = 1;\ V(s_3) = 0;\ V(s_5) = 10;\ V(s_6) = 0$
  - ★ Third trajectory $(s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6)$
    $V(s_1) = 1;\ V(s_3) = 0.33;\ V(s_4) = 1;\ V(s_6) = 0$
  - ★ Fourth trajectory $(s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6)$
    $V(s_1) = 1.08;\ V(s_3) = 0.5;\ V(s_4) = 1;\ V(s_6) = 0$
  - ★ Fifth trajectory $(s_2 \xrightarrow{2} s_3 \xrightarrow{0} s_5 \xrightarrow{10} s_6)$
    $V(s_2) = 2.5;\ V(s_3) = 2.4;\ V(s_5) = 10;\ V(s_6) = 0$

TD Algorithms: A Schematic View



- Uses experience without model like MC
- Bootstraps like DP
- Can work with partial sequences
- Suited for online learning

Schematic View of Various Algorithms

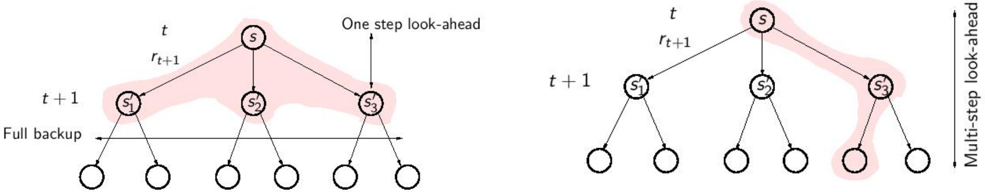
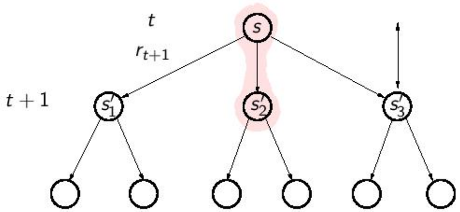


Figure: DP Algorithm and MC Algorithm



Convergence of TD Algorithms

- For any fixed policy π, the TD(0) algorithm described above converges (asymptotically) to V^{π} under some conditions on the choice of α (Robbins Monroe Condition)
  - ★ $\sum \alpha_t = \infty$
  - ★ $\sum \alpha_t^2 < \infty$

- *Generally*, TD methods have usually been found to converge faster than MC methods on certain class of tasks