



AIL 7022: Reinforcement Learning

Lecture 10: Temporal Difference Prediction

Instructor: Raunak Bhattacharyya



ScAI

YARDI SCHOOL OF ARTIFICIAL INTELLIGENCE
INDIAN INSTITUTE OF TECHNOLOGY DELHI

Policy Iteration

$$\pi_0 \xrightarrow{\text{E}} v_{\pi_0} \xrightarrow{\text{I}} \pi_1 \xrightarrow{\text{E}} v_{\pi_1} \xrightarrow{\text{I}} \pi_2 \xrightarrow{\text{E}} \cdots \xrightarrow{\text{I}} \pi_* \xrightarrow{\text{E}} v_*$$

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization

$V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$; $V(\text{terminal}) \doteq 0$

2. Policy Evaluation

Loop:

$\Delta \leftarrow 0$

Loop for each $s \in \mathcal{S}$:

$v \leftarrow V(s)$

$V(s) \leftarrow \sum_{s',r} p(s', r | s, \pi(s)) [r + \gamma V(s')]$

$\Delta \leftarrow \max(\Delta, |v - V(s)|)$

until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement

$\text{policy-stable} \leftarrow \text{true}$

For each $s \in \mathcal{S}$:

$\text{old-action} \leftarrow \pi(s)$

$\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r | s, a) [r + \gamma V(s')]$

If $\text{old-action} \neq \pi(s)$, then $\text{policy-stable} \leftarrow \text{false}$

If policy-stable , then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Greedy Policy



	1	2	3
4	5	6	7
8	9	10	11
12	13	14	

$R_t = -1$
on all transitions

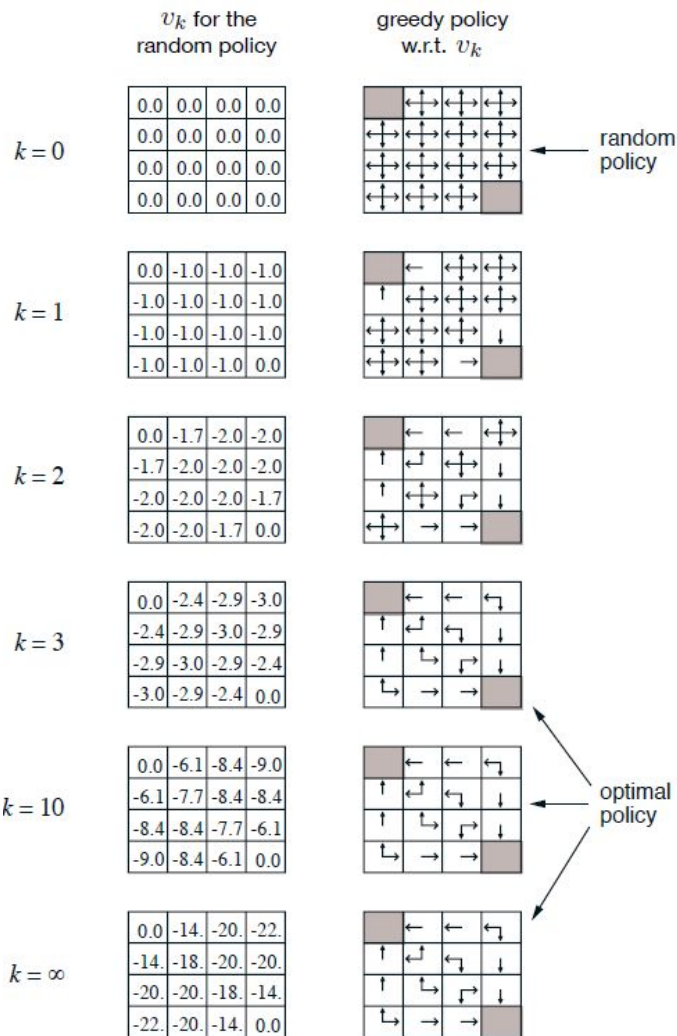
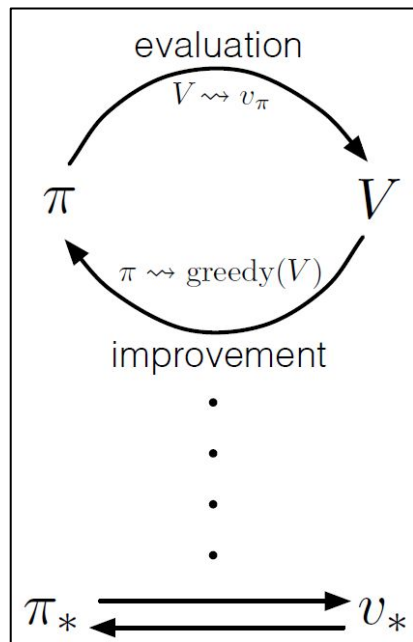
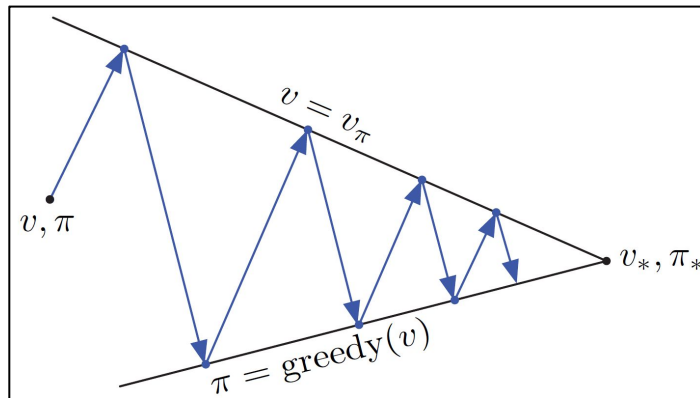


Fig 4.1, [Sutton & Barto](#)

Generalised Policy Iteration

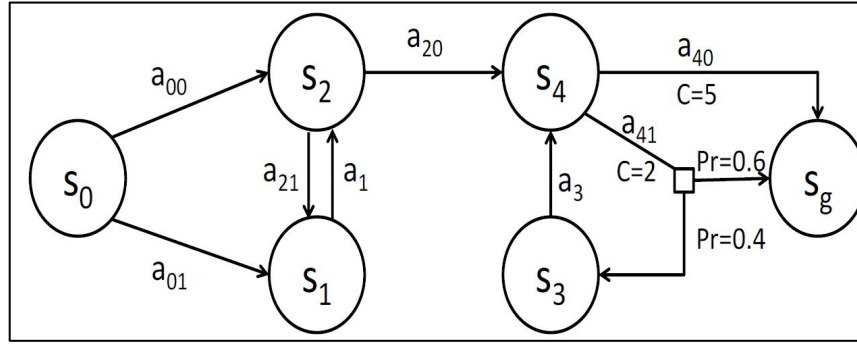


- Two simultaneous, interacting processes
 - Make value fun consistent with current policy
 - Make policy greedy w.r.t. current value function



GPI: Evaluation and improvement processes interact, independent of granularity

Value Iteration



n	$V_n(s_0)$	$V_n(s_1)$	$V_n(s_2)$	$V_n(s_3)$	$V_n(s_4)$
0	3	3	2	2	1
1	3	3	2	2	2.8
2	3	3	3.8	3.8	2.8
3	4	4.8	3.8	3.8	3.52
4	4.8	4.8	4.52	4.52	3.52
5	5.52	5.52	4.52	4.52	3.808
10	5.9232	5.9232	4.96928	4.96928	3.96928
20	5.99921	5.99921	4.99969	4.99969	3.99969

Demo: Value Iteration

GridWorld: Dynamic Programming Demo

Policy Evaluation (one sweep)	Policy Update	Toggle Value Iteration	Reset
-------------------------------	---------------	------------------------	-------

0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖
0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖
0.00 ↖					0.00 ↖				0.00 ↖
0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖ R -1.0		0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖
0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖		0.00 ↖ R -1.0	0.00 ↖ R -1.0	0.00 ↖	0.00 ↖	0.00 ↖
0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖		0.00 ↖ R 1.0	0.00 ↖ R -1.0	0.00 ↖	0.00 ↖ R -1.0	0.00 ↖
0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖		0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖ R -1.0	0.00 ↖
0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖ R -1.0		0.00 ↖ R -1.0	0.00 ↖ R -1.0	0.00 ↖	0.00 ↖	0.00 ↖
0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖	0.00 ↖

DP: Summary

Problem	Bellman Equation	Algorithm
Prediction	Bellman Expectation Equation	Iterative Policy Evaluation
Control	Bellman Expectation Equation + Greedy Policy Improvement	Policy Iteration
Control	Bellman Optimality Equation	Value Iteration

Why don't we do the above things with q-values?

Model-Free Prediction

Model-Free Approaches

- MDP model is unknown, but we can sample from it

**Autonomous vehicle in
traffic**

**Robotic navigation in
unknown envs**

**Advertising with
unknown user behavior**

- MDP model is known, but it's easier to sample

Climate models

Robotic navigation

Game playing

Monte Carlo Estimation

Setting

Episodes and visits

$$\mathbb{E}[f(x)] \quad \hat{\mu} = \frac{1}{N} \sum_{i=1}^N f(x_i)$$

where $x_1, x_2, \dots, x_N$ are i.i.d. random samples drawn from the distribution over x .

- Sample trajectories
- Store the obtained returns
- Average

Temporal Difference

- Can learn from incomplete episodes
- Setting: Learn online, i.e. after every timestep

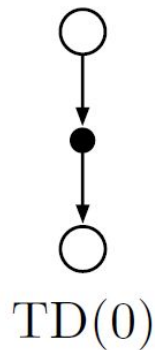
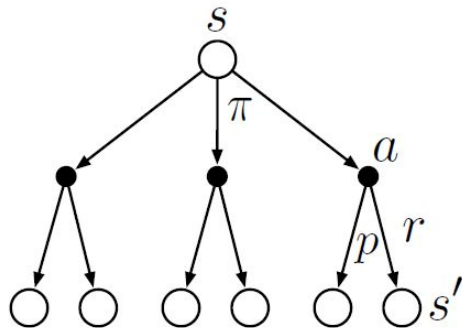


Temporal Difference Policy Evaluation

No reality, bootstrap

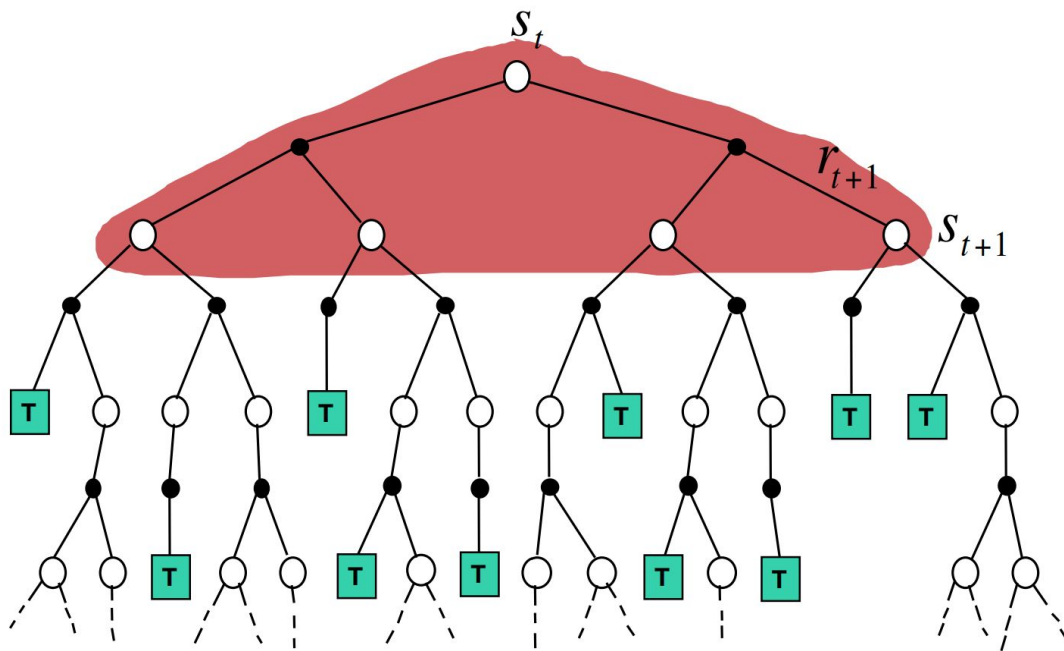
$$V(S_t) \leftarrow V(S_t) + \alpha (R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$$

No expectation, single sample



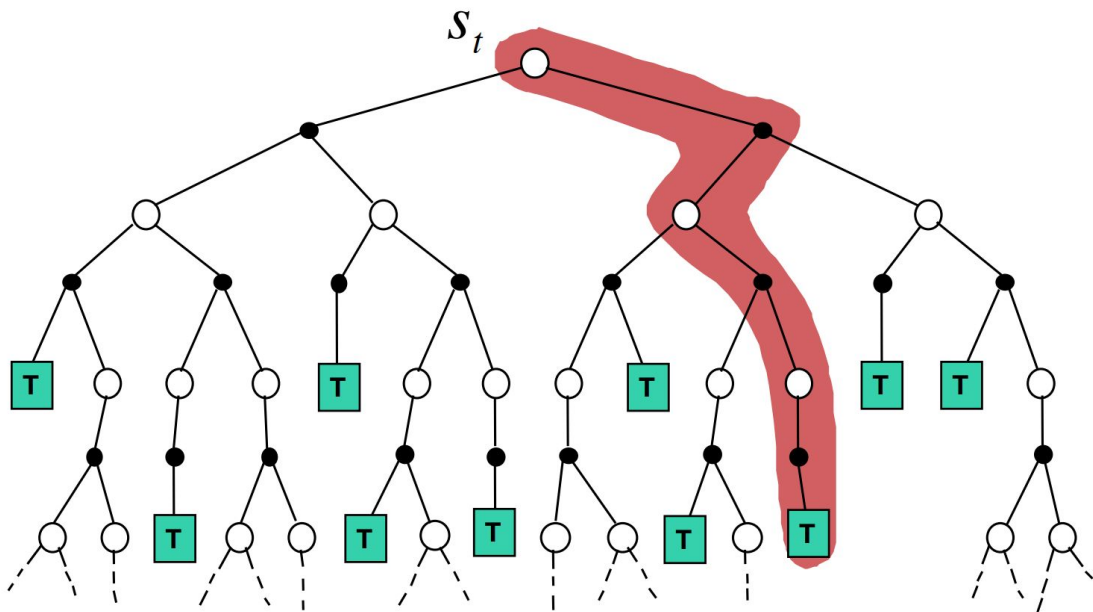
Visual: Dynamic Programming

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



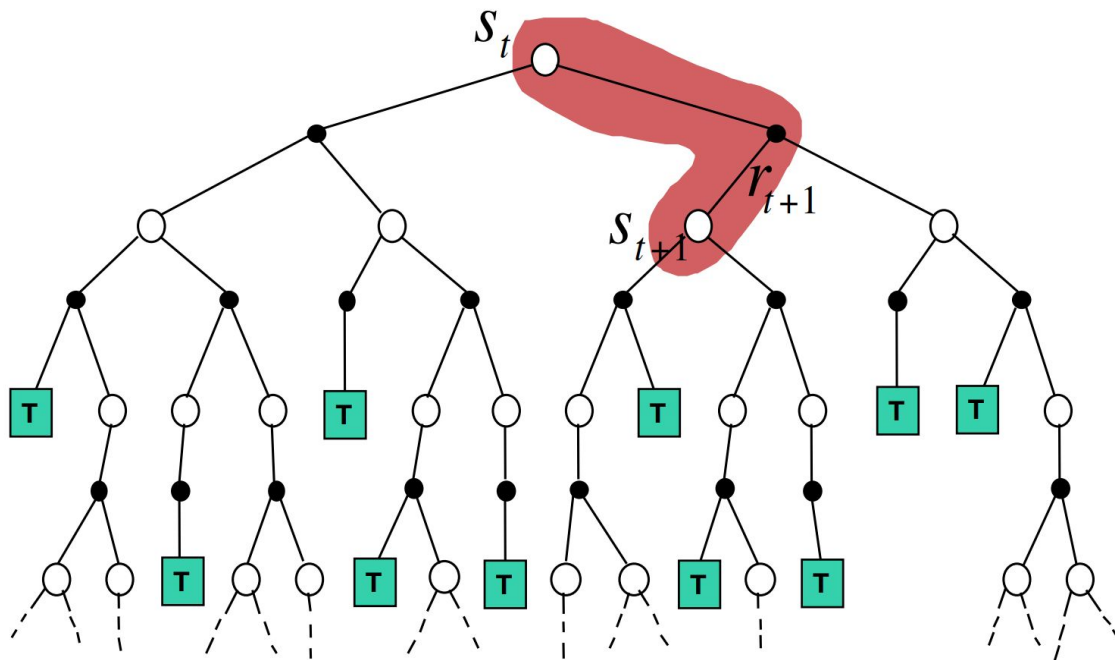
Visual: Monte-Carlo

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



Visual: Temporal Difference

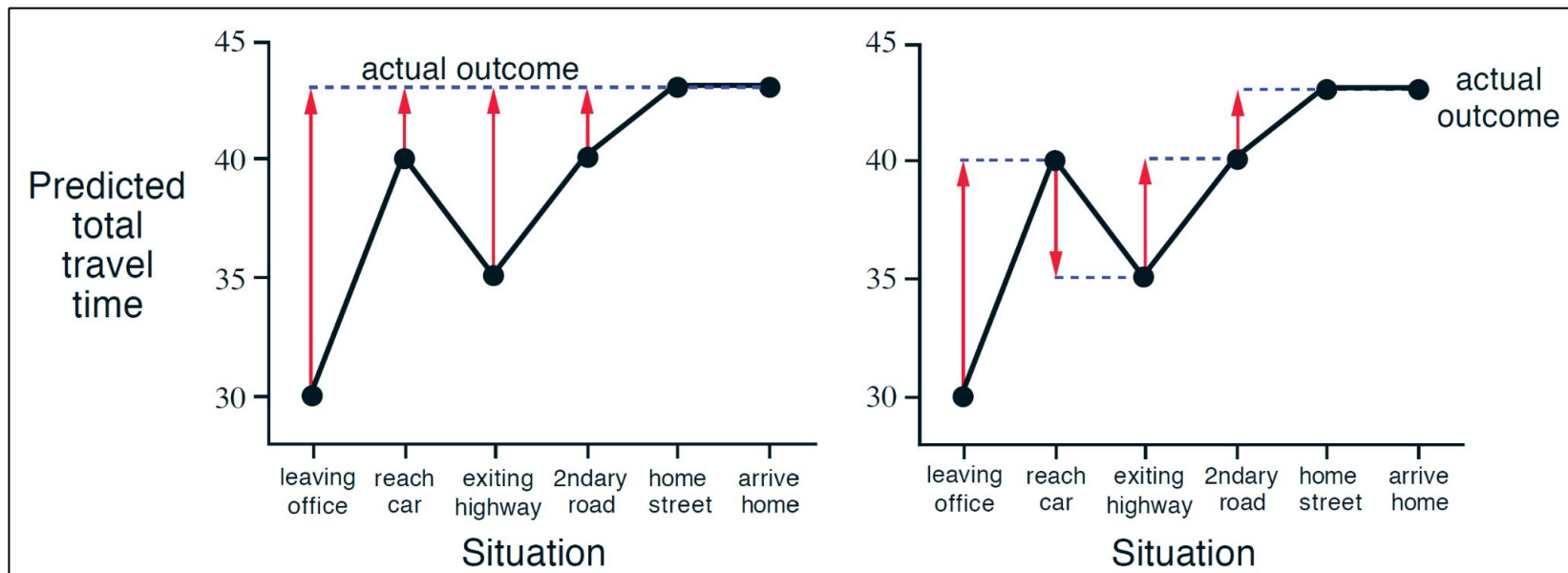
$$V(S_t) \leftarrow V(S_t) + \alpha (R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$$



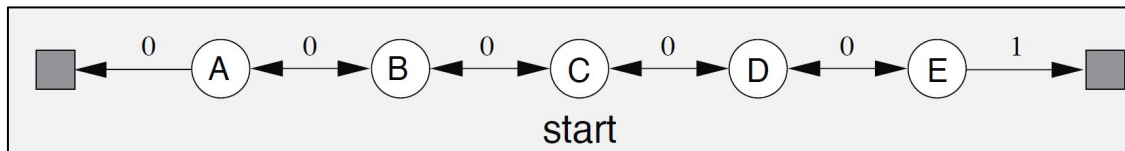
TD: Driving Example

<i>State</i>	<i>Elapsed Time (minutes)</i>	<i>Predicted Time to Go</i>	<i>Predicted Total Time</i>
leaving office, friday at 6	0	30	30
reach car, raining	5	35	40
exiting highway	20	15	35
2ndary road, behind truck	30	10	40
entering home street	40	3	43
arrive home	43	0	43

TD: Driving Example



Random Walk



Tabular TD(0) for estimating v_π

Input: the policy π to be evaluated

Algorithm parameter: step size $\alpha \in (0, 1]$

Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\text{terminal}) = 0$

Loop for each episode:

 Initialize S

 Loop for each step of episode:

$A \leftarrow$ action given by π for S

 Take action A , observe R, S'

$V(S) \leftarrow V(S) + \alpha [R + \gamma V(S') - V(S)]$

$S \leftarrow S'$

 until S is terminal

Random Walk

