In the Name of God

Iran University of Science and Technology

Reinforcement Learning in Control

Dr. Saeed Shamaghdari

Electrical Engineering Department
Control Group

Fall 2025 | 4041

Temporal Difference Learning

Temporal Difference Learning

Q Box:
Similarity to dynamic programming (DP)?
Similarity to Monte Carlo methods (MCs)?

Reminder Box

Value function approximation in DP and MC

$$\underbrace{V(S_t)}_{\text{New value of state t}} \leftarrow \underbrace{V(S_t)}_{\text{Former estimation of value of state t (= Expected return starting at that state)}} + \underbrace{\alpha}_{\text{Learning Rate}}[\underbrace{G_t}_{\text{Return at timestep t}} - \underbrace{V(S_t)}_{\text{Former estimation of value of state t (= Expected return starting at that state)}}]$$

The Constant-alpha method in MC (requires episode completion)
Using Return as an estimate of the expected value (Sampling)

Temporal Difference Prediction

Value function approximation TD

$$\underbrace{V(S_t)}_{\text{New value of state t}} \leftarrow \underbrace{V(S_t)}_{\text{Former estimation of value of state t}} + \underbrace{\alpha}_{\text{Learning Rate}}[\underbrace{R_{t+1}}_{\text{Reward}} + \underbrace{\gamma V(S_{t+1})}_{\text{Discounted value of next state}} - V(S_t)]$$

Q Box:

Target in MC and TD?

Temporal Difference Prediction

Value function approximation TD

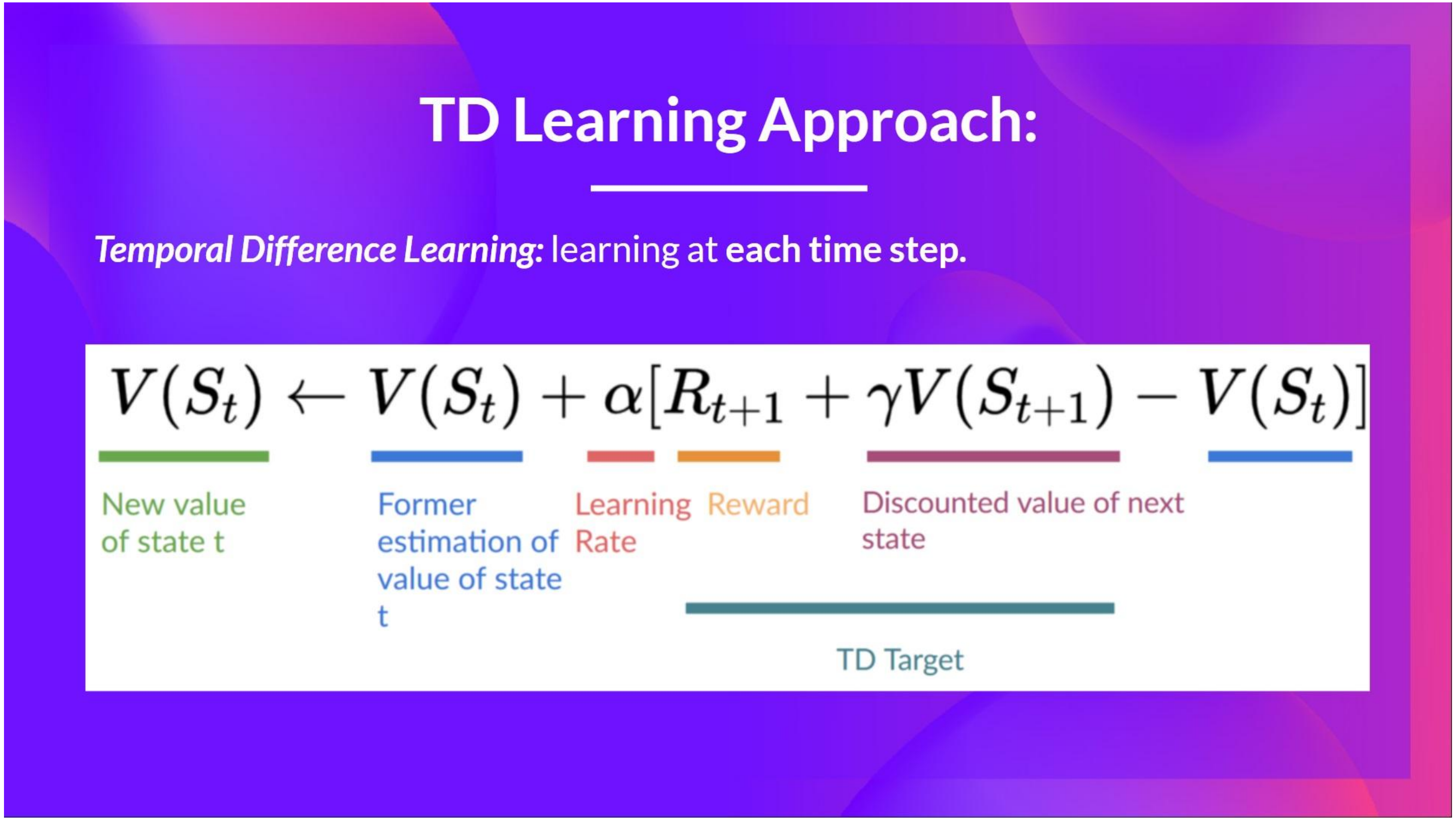
$$\underbrace{V(S_t)}_{\text{New value of state t}} \leftarrow \underbrace{V(S_t)}_{\text{Former estimation of value of state t}} + \underbrace{\alpha}_{\text{Learning Rate}}[\underbrace{\underbrace{R_{t+1}}_{\text{Reward}} + \underbrace{\gamma V(S_{t+1})}_{\text{Discounted value of next state}}}_{\text{TD Target}} - \underbrace{V(S_t)}]$$

TD(0): one step TD

Q Box:

Target in MC and TD?

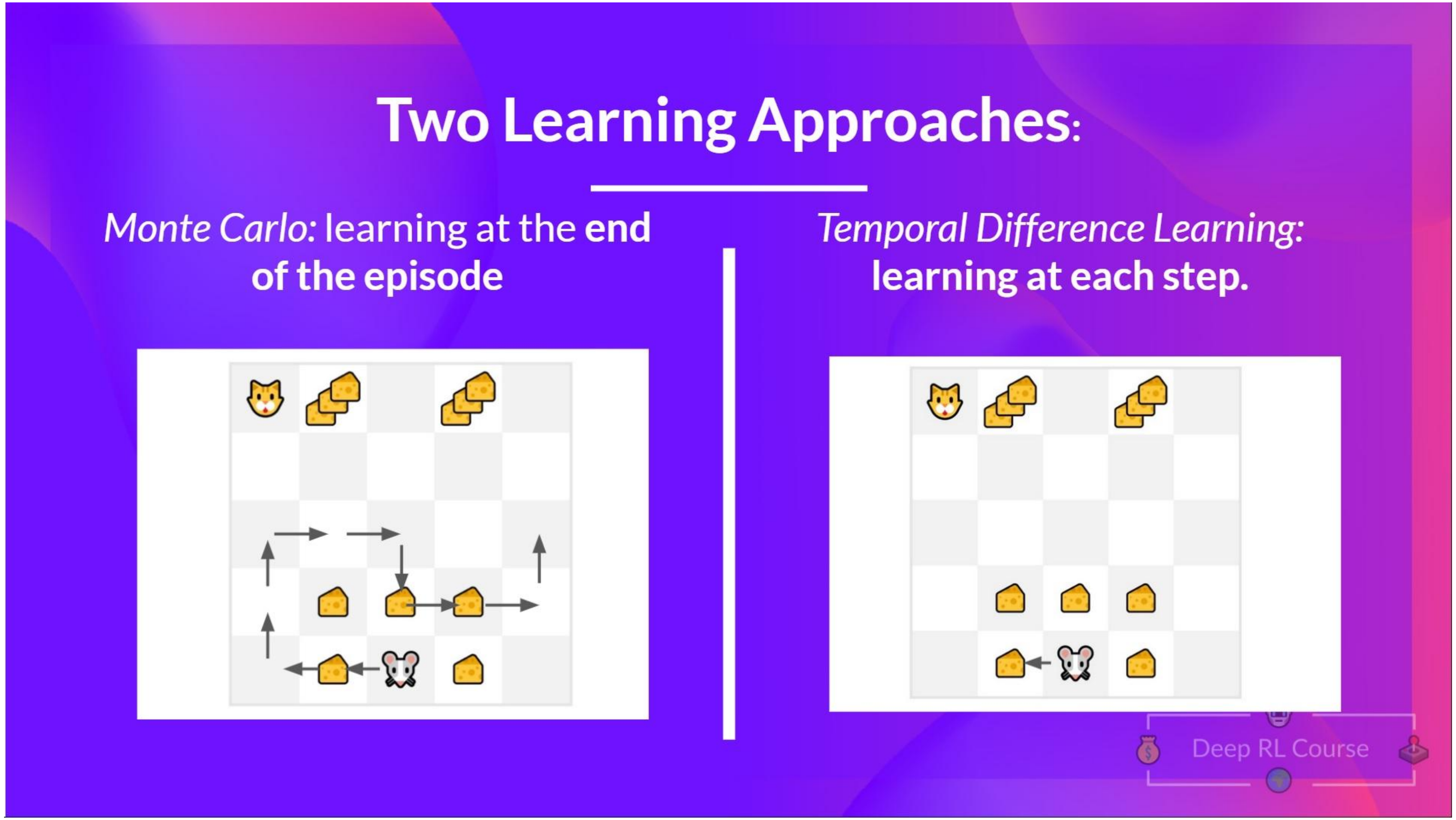
To Recap …


TD Learning Approach:
Temporal Difference Learning: learning at **each time step.**
$V(S_t) \leftarrow V(S_t) + \alpha[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$
New value of state t
Former estimation of value of state t
Learning Rate
Reward
Discounted value of next state
TD Target

To Recap ...


TD Approach:
At the end of one step (State, Action, Reward, Next State):
- We have Rt+1 and St+1
- We update V(St):
- We estimate Gt by adding Rt+1 and the discounted value of next state.
TD target : Rt+1 + gamma * V(St+1)
$V(S_t) \leftarrow V(S_t) + \alpha[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$
Now we continue to interact with this environment with our updated value function.
By running more and more steps, the agent will learn to play better and better.

To Recap …


Two Learning Approaches:
Monte Carlo: learning at the end of the episode
Temporal Difference Learning: learning at each step.
Deep RL Course

Temporal Difference Prediction

Value function approximation TD

$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s]
\end{aligned}
$$

- **DP:** Computing the expected value using the model, utilizing previous value function estimates in new estimates (bootstrapping).
- **TD:** Sampling, using previous value function estimates in new estimates (bootstrapping).

Algorithm

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 $V(terminal) = 0$

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'
    $V(S) \leftarrow V(S) + \alpha\big[R + \gamma V(S') - V(S)\big]$
    $S \leftarrow S'$
  until S is terminal

TD Error

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

Monte Carlo Error

$$\begin{aligned}
G_t - V(S_t) &= R_{t+1} + \gamma G_{t+1} - V(S_t) + \gamma V(S_{t+1}) - \gamma V(S_{t+1}) \\
&= \delta_t + \gamma\big(G_{t+1} - V(S_{t+1})\big) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2\big(G_{t+2} - V(S_{t+2})\big) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2} + \cdots + \gamma^{T-t-1}\delta_{T-1} + \gamma^{T-t}\big(G_T - V(S_T)\big) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2\delta_{t+2} + \cdots + \gamma^{T-t-1}\delta_{T-1} + \gamma^{T-t}(0 - 0) \\
&= \sum_{k=t}^{T-1} \gamma^{k-t}\delta_k.
\end{aligned}$$

Example: Driving Home

Value function approximation TD

State	*Elapsed Time (minutes)*	*Predicted Time to Go*	*Predicted Total Time*
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

Q Box

Reward?
Return? T_Go
Value? E(T_Go)

Example: Driving Home

Value function approximation TD

State	*Elapsed Time (minutes)*	*Predicted Time to Go*	*Predicted Total Time*
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

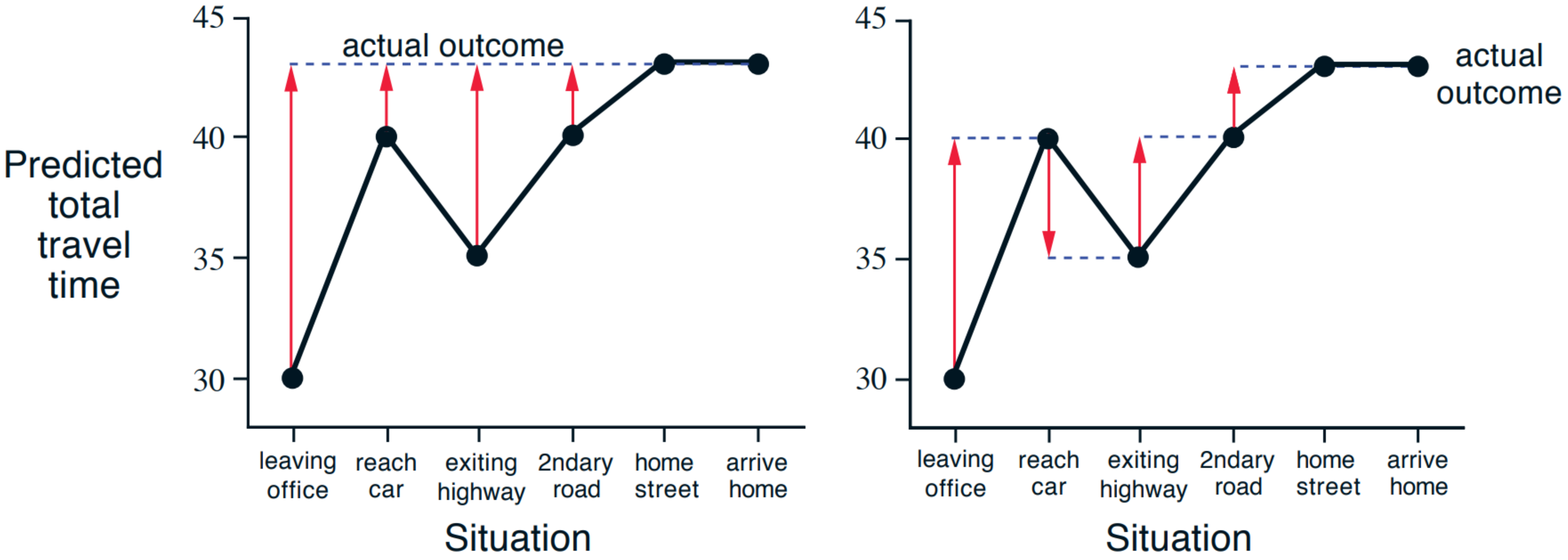
Q Box

Estimation error at highway exit (based on Monte Carlo)?

Estimation correction at highway exit (MC-based) for alpha=0.5?

$$\alpha\big(G_t - V(s_t)\big)?$$

TD: Faster Learning



Temporal Difference Learning

TD Convergence

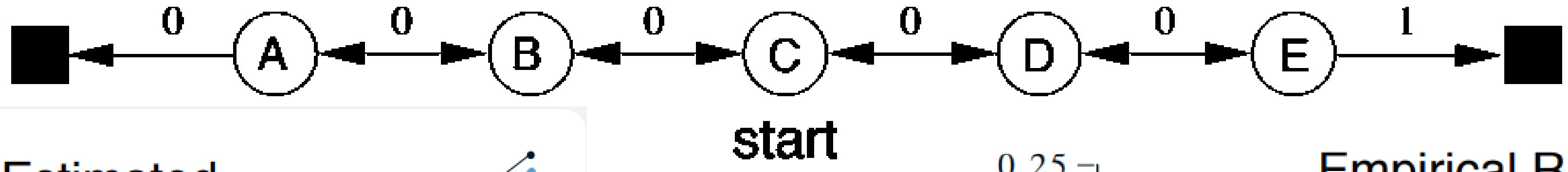
- Very small α
- α with decay conditions

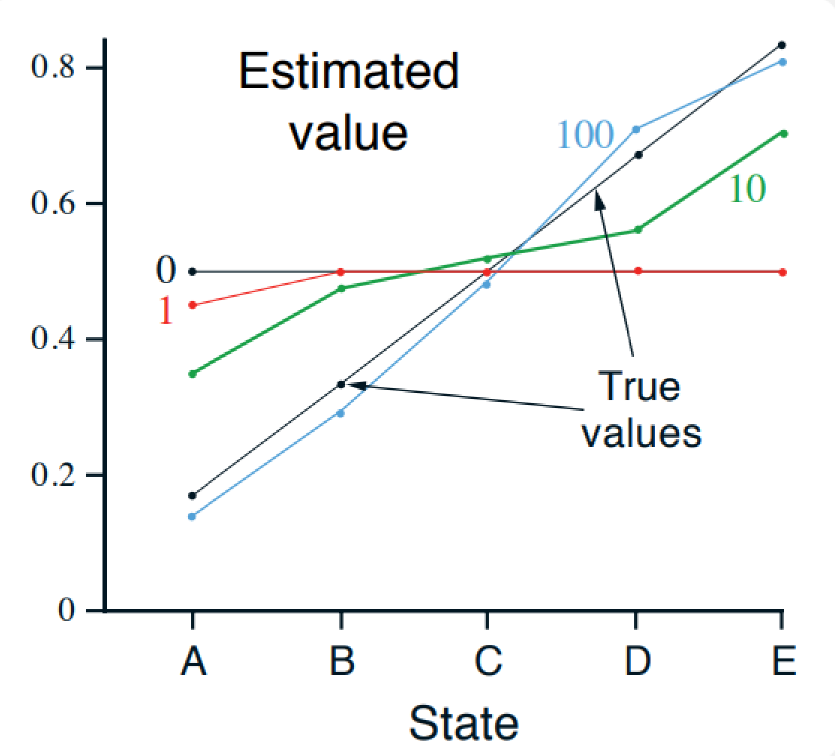
$$\sum_{n=1}^{\infty} \alpha_n(a) = \infty \qquad \text{and} \qquad \sum_{n=1}^{\infty} \alpha_n^2(a) < \infty$$

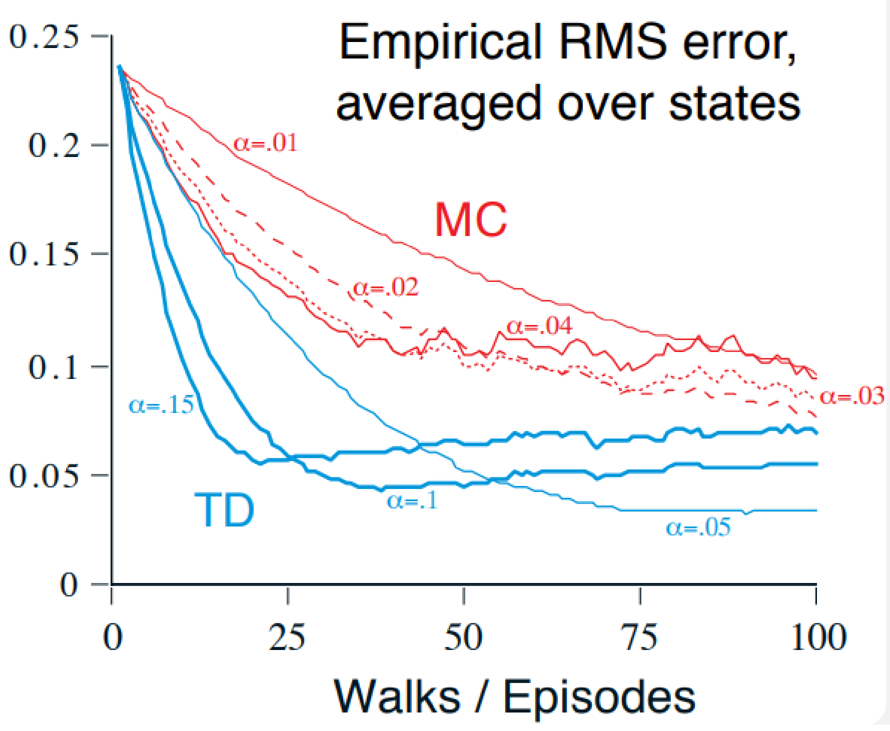
Convergence speed of TD and MC

Markov Reward Process: Random Walk Example

In this example we empirically compare the prediction abilities of TD(0) and constant-α MC when applied to the following Markov reward process:







Random Walk Under Batch Updating

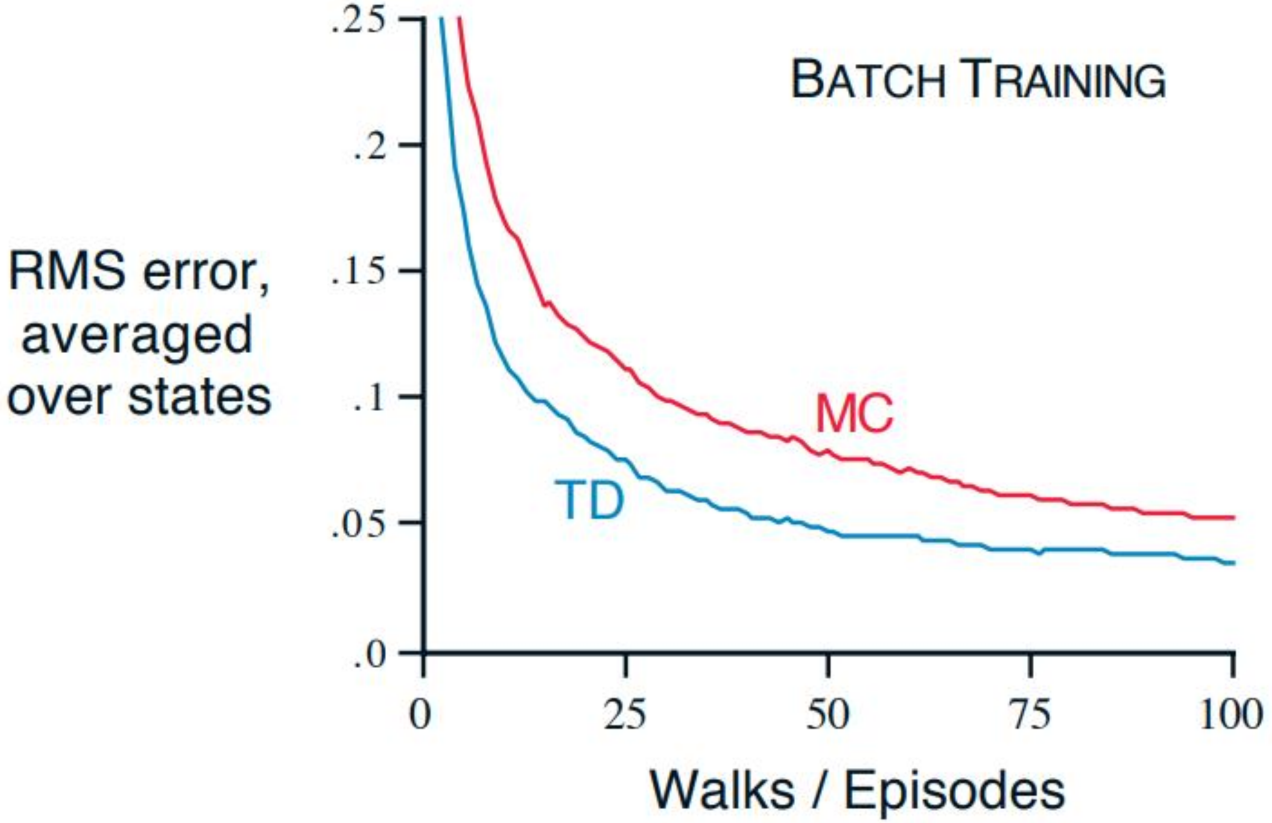
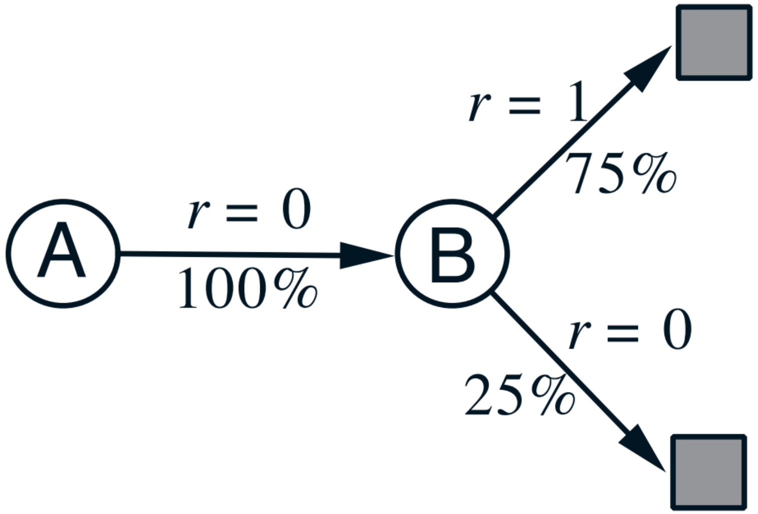


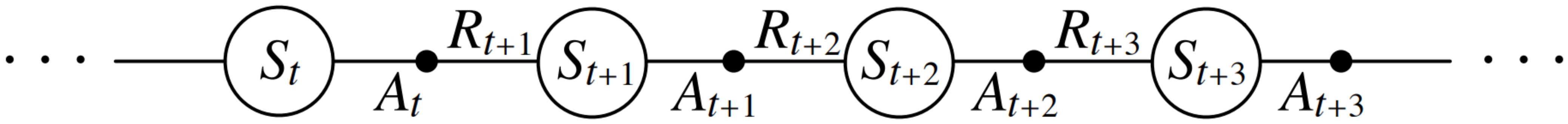
Figure 6.2: Performance of TD(0) and constant-α MC under batch training on the random walk task.

Example: You are the Predictor

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

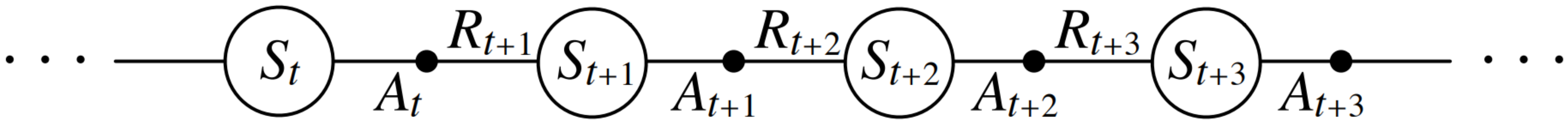


SARSA: On-Policy TD Control



$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\Big[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\Big]$$

SARSA: On-Policy TD Control



$$\underbrace{Q(S_t, A_t)}_{\text{Updated Q Value}} \leftarrow \underbrace{Q(S_t, A_t)}_{\text{Current Q Value}} + \alpha \Big[\overbrace{R_{t+1} + \underbrace{\gamma Q(S_{t+1}, A_{t+1})}_{\text{Target Policy is always as same as Behaviour Policy}}}^{\text{Target Q Value}} - \underbrace{Q(S_t, A_t)}_{\text{Current Q Value}} \Big]$$

Algorithm

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
    Initialize S
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Loop for each step of episode:
        Take action A, observe R, S'
        Choose A' from S' using policy derived from Q (e.g., ε-greedy)
        $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') - Q(S, A)\big]$
        $S \leftarrow S'$; $A \leftarrow A'$;
    until S is terminal

Better Intuition

Select Action

Agent chooses action based on policy

Observe State

Agent perceives current environment

Update Q-value

Agent adjusts Q-values based on experience

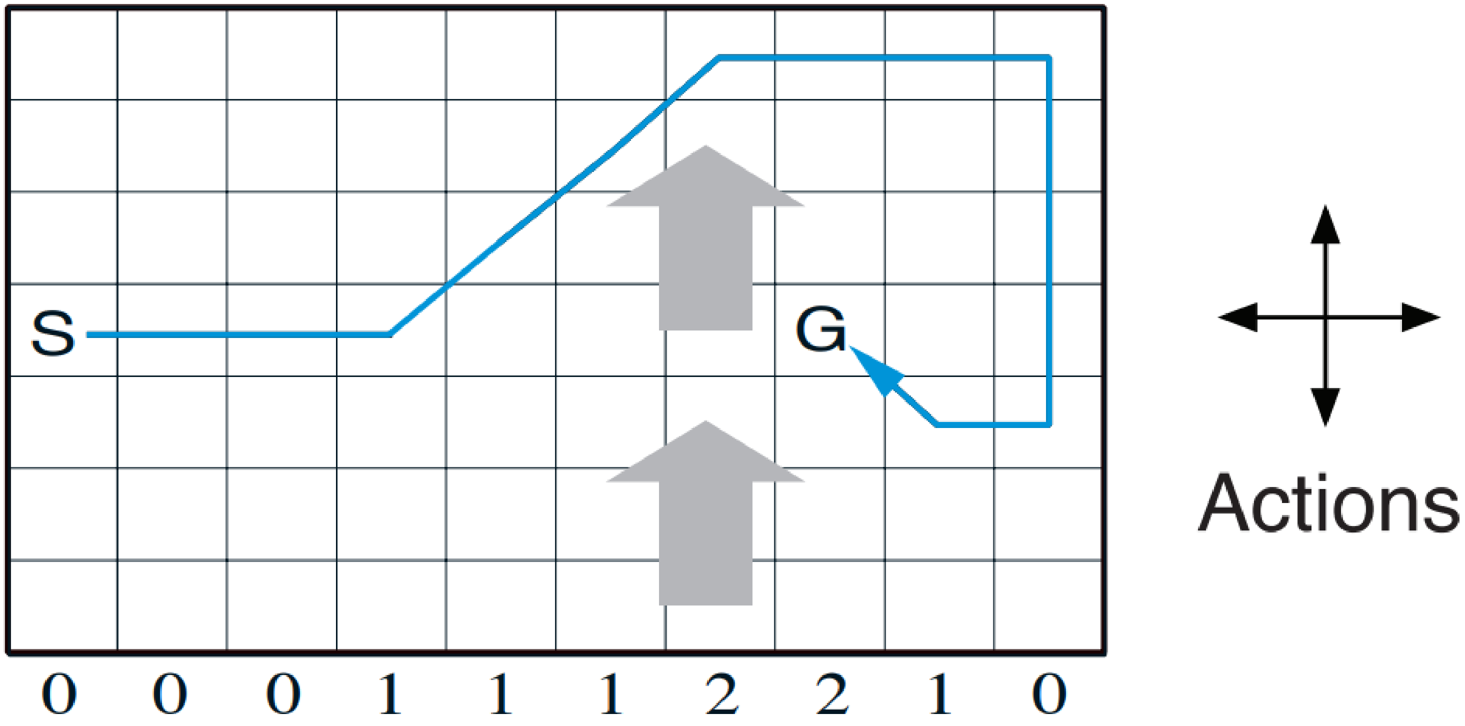
Execute Action !

Agent performs selected action

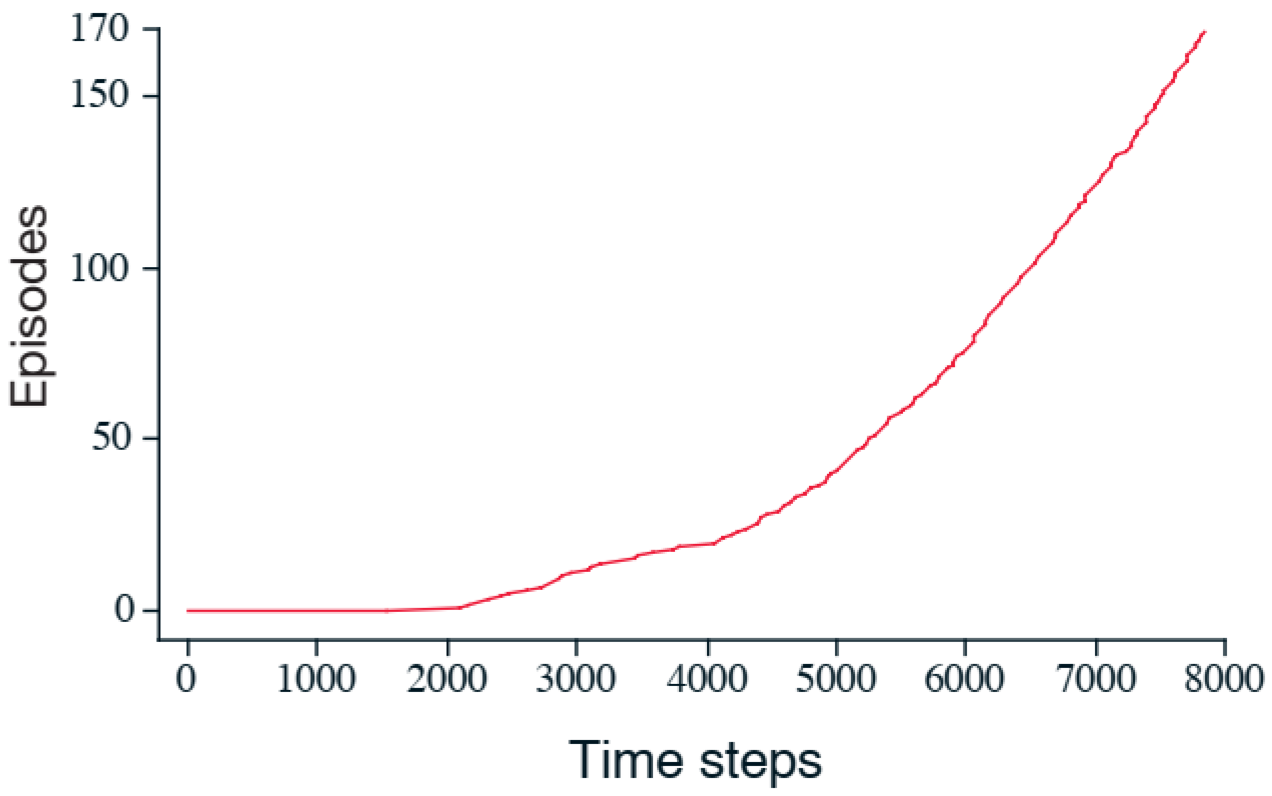
Observe Reward

Agent receives feedback from environment

Example: Windy Grid



Example: Windy Grid



Q-Learning: Off-Policy TD Control

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma max_a Q(S_{t+1}, a) - Q(S_t, A_t)]$$

New Q-value estimation

Former Q-value estimation

Learning Rate

Immediate Reward

Discounted Estimate optimal Q-value of next state

Former Q-value estimation

TD Target

TD Error

Example

- You always start at the **same starting point.**
- The goal: eat the **big pile of cheese** (at the bottom right-hand corner) and **avoid the poison.**
- The episode ends if we eat the poison, eat the big pile of cheese or if we spent more than 5 steps.
- Learning rate = 0.1
- Gamma = 0.99

Example

- The reward function:
  - 0: Going to a state **with no cheese in it.**
  - +1: Going to a state with a **small cheese in it.**
  - +10: Going to the state with **the big pile of cheese.**
  - -10: Going to the state **with the poison and thus die.**

Example, Step 1

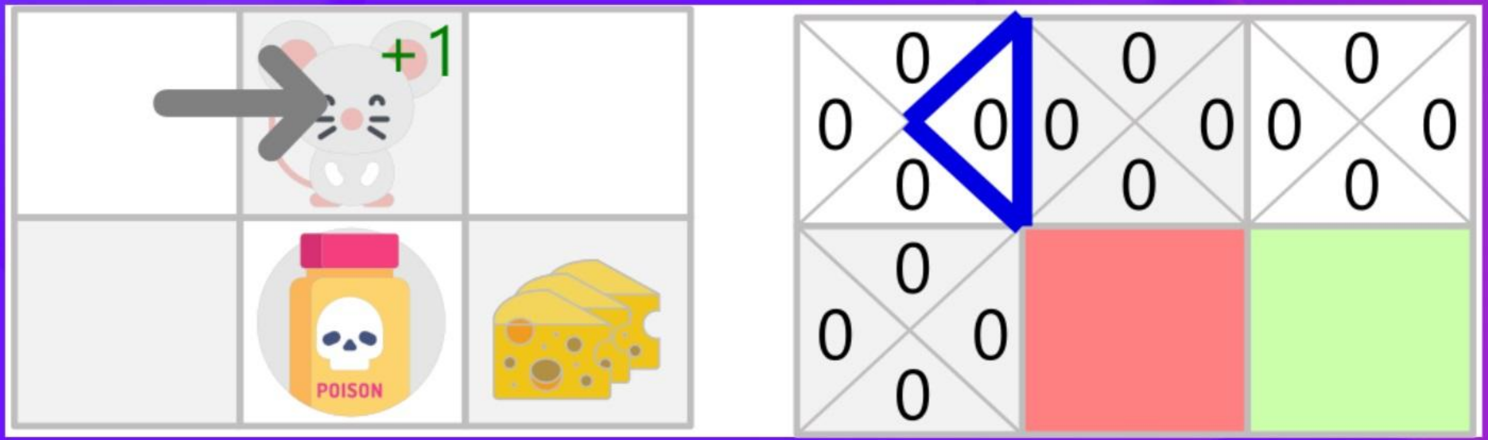
Initialize Q arbitrarily (e.g., $Q(s,a) = 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$, and $Q(\textit{terminal-state}, \cdot) = 0$)

	←	→	↑	↓
	0	0	0	0
	0	0	0	0
	0	0	0	0
	0	0	0	0
	0	0	0	0
	0	0	0	0

We initialize the Q-Table

Example, Step 2

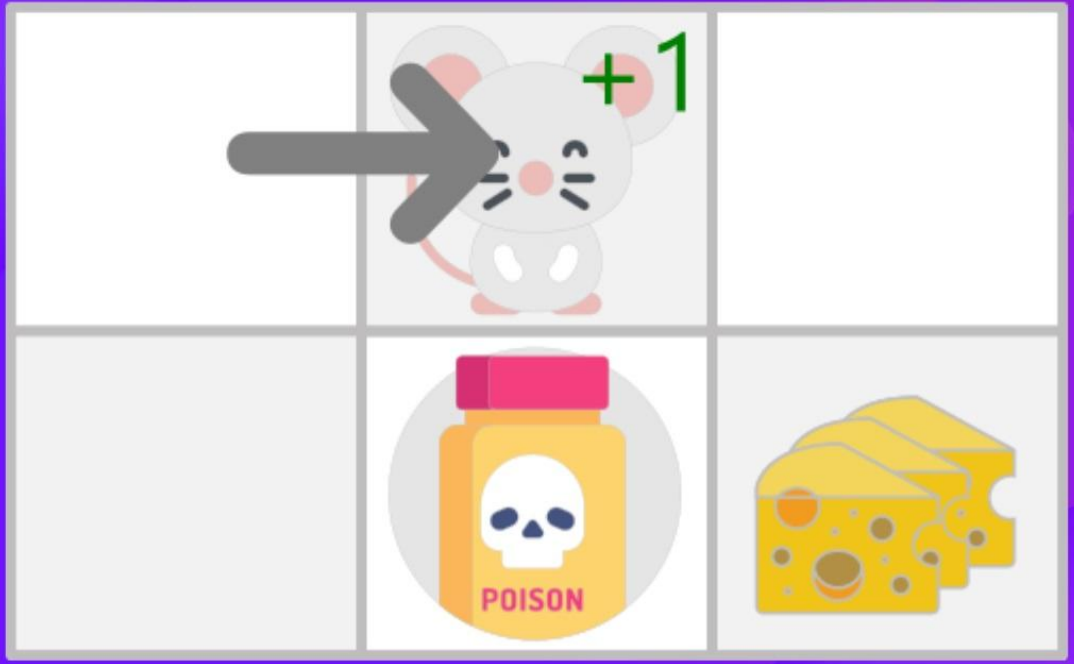
Choose action A_t using policy derived from Q (e.g., ϵ-greedy)

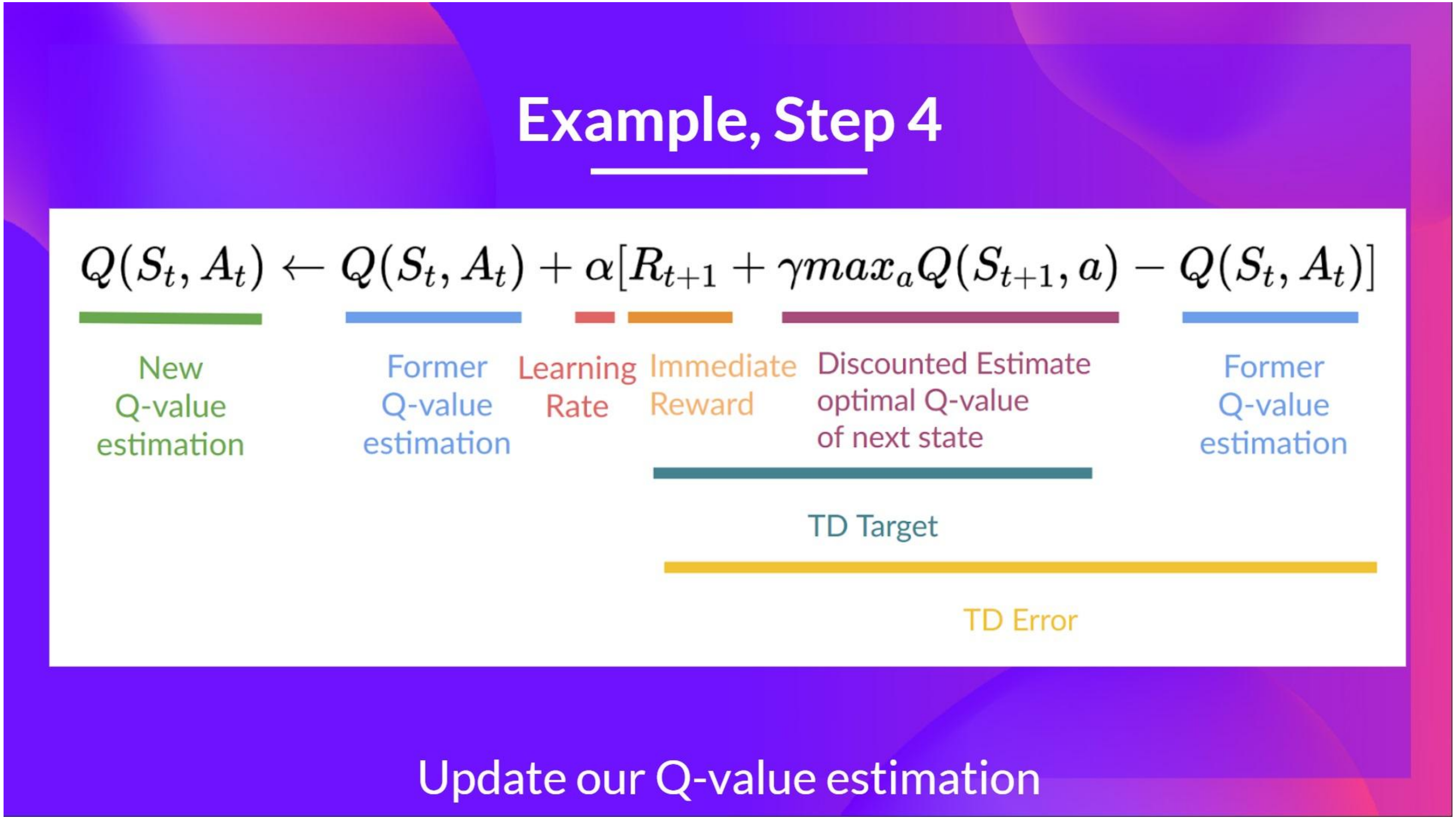


We took a random action (exploration)

Example, Step 3

Take action A_t and observe R_{t+1}, S_{t+1}




Example, Step 4
$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma max_a Q(S_{t+1}, a) - Q(S_t, A_t)]$
New Q-value estimation
Former Q-value estimation
Learning Rate
Immediate Reward
Discounted Estimate optimal Q-value of next state
Former Q-value estimation
TD Target
TD Error
Update our Q-value estimation

Example, Step 4

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$$

Q(Initial state, Right) = 0 + 0.1 * [1 + 0.99 * 0 - 0]
Q(Initial state, Right) = 0.1

	←	→	↑	↓
	0	0.1	0	0
	0	0	0	0
	0	0	0	0
	0	0	0	0
	0	0	0	0
	0	0	0	0

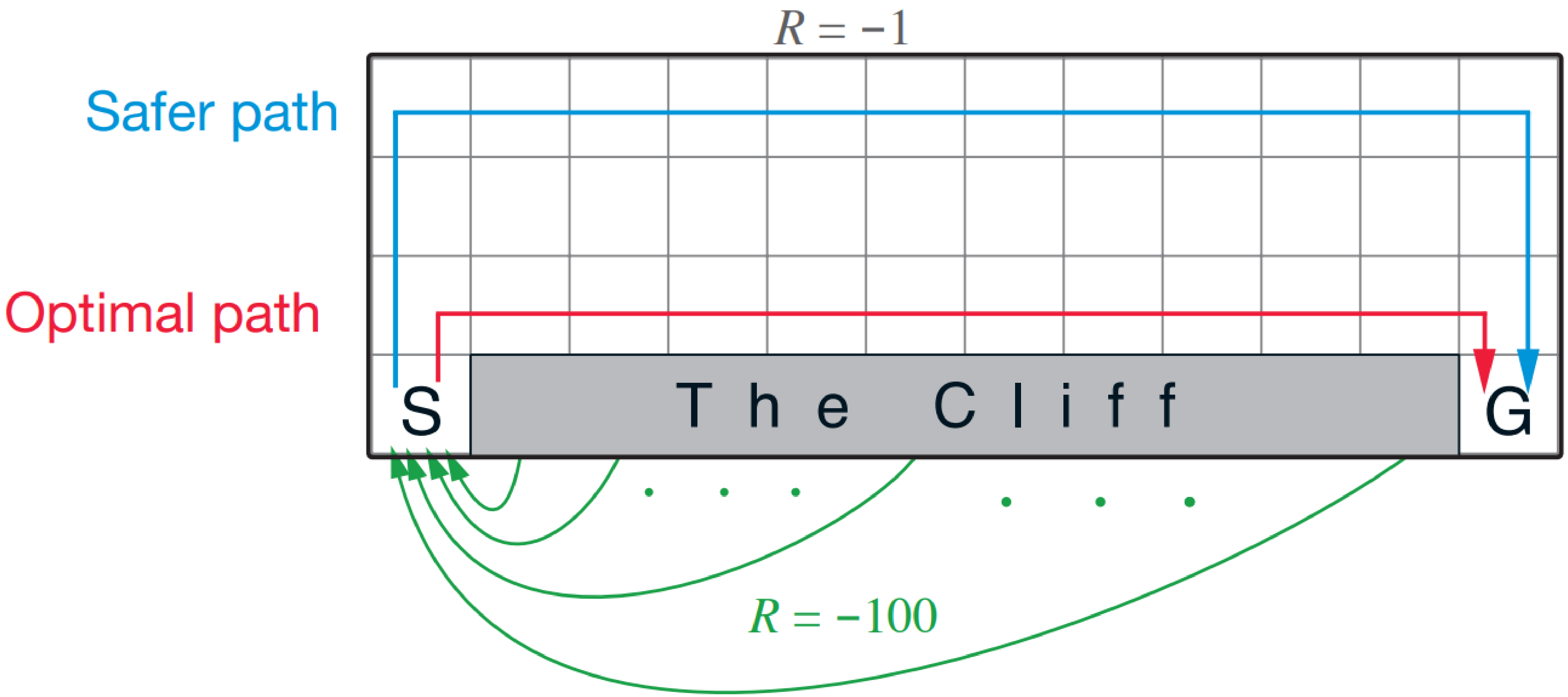
Q-Learning Recap: Algorithm

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\left[R + \gamma \max_a Q(S', a) - Q(S, A)\right]$
    $S \leftarrow S'$
  until S is terminal

Example: Cliff Walking



Example: Cliff Walking