Fraunhofer IIS

Fraunhofer-Institut für Integrierte Schaltungen IIS

# Reinforcement Learning

—

# Exercise 3: Model-free Control, SARSA & Q-Learning

Nico Meyer

# Exercise Sheet 3

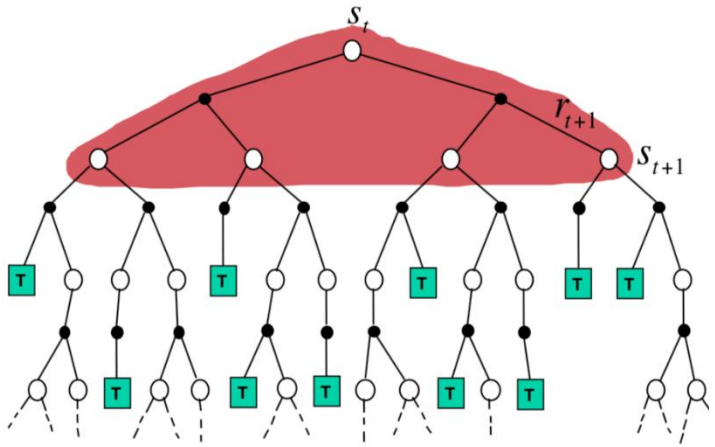
## Gymnasium Gridworld

# Exercise Sheet 3

## Temporal Difference Learning

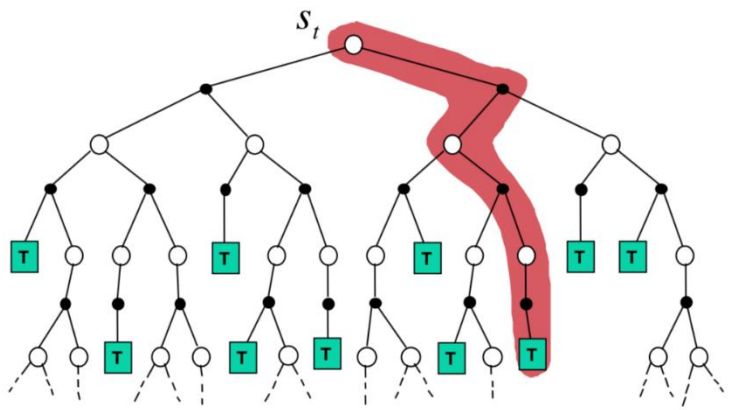
DP Backup

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



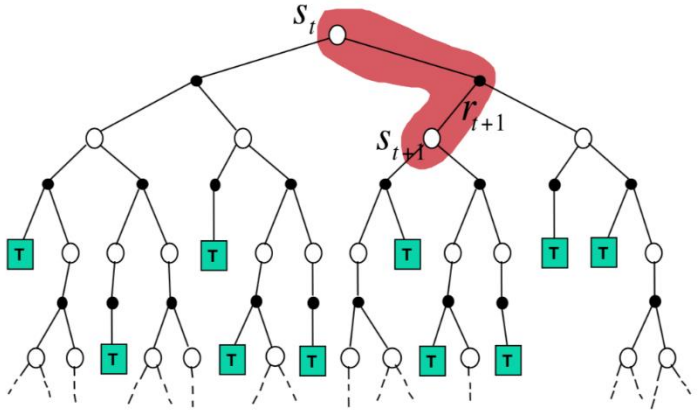

MC Backup

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




TD Backup

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




*Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.*

# Model-free Control

## TD Methods

# Model-free Prediction vs. Control

**Problem:**

We do not know $\mathcal{P}$ or $\mathcal{R}$ or both of the MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$

**Solution:**

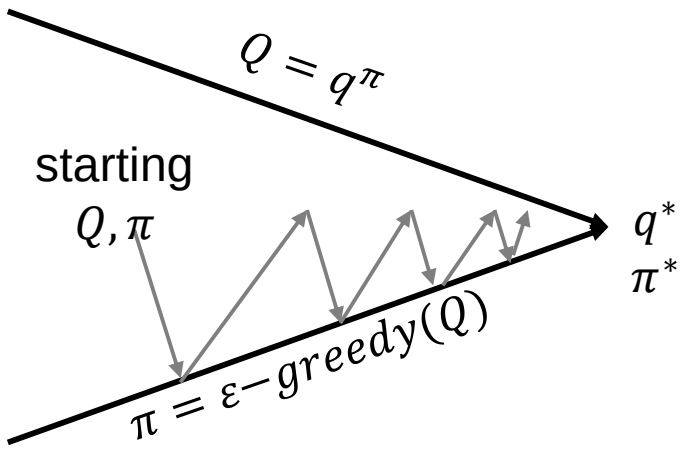
Model-free methods that use experience samples $\mathrm{s}(s, a, r, s')$

**In Exercise 3 we did:**

**Model-free Prediction:** Evaluate the future, given the policy $\pi$.
*(estimate the value function)*

**In Exercise 4 we will do:**

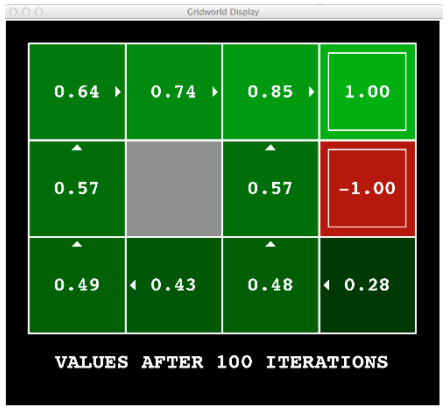
**Model-free Control:** Optimize the future by finding the best policy $\pi$.
*(optimize the value function)*

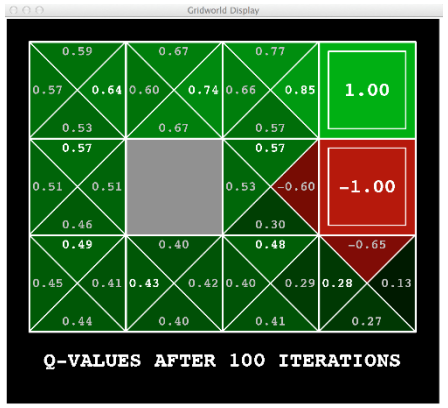


# State-action-value function

$$s \xrightarrow{a, r_0} s_1 \xrightarrow{\pi(S_1), r_1} s_2 \xrightarrow{\pi(S_2), r_2} s_3 \dots \quad s_{h-1} \xrightarrow{\pi(s_{h-1}), r_{h-1}} s_h$$

$$Q^{\pi}(s, a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a\right]$$




**Greedy Policy Improvement over Q**:

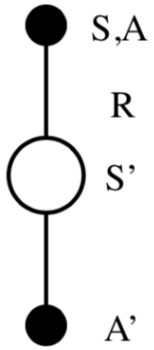
$$\pi'(s) = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\, Q^{\pi}(s, a)$$

$$\forall s \in \mathcal{S}, \qquad Q^{\pi'}\big(s, \pi'(s)\big) \geq Q^{\pi}\big(s, \pi(s)\big)$$

# SARSA

## on-policy control

- Apply TD to $Q(s,a)$
- Use $\varepsilon$-greedy policy improvement
- Update at every time-step




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

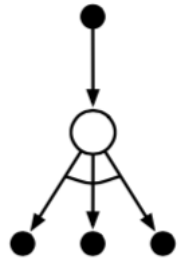
Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize $S$
  Choose $A$ from $S$ using policy derived from $Q$ (e.g., $\varepsilon$-greedy)
  Loop for each step of episode:
    Take action $A$, observe $R$, $S'$
    Choose $A'$ from $S'$ using policy derived from $Q$ (e.g., $\varepsilon$-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

*Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.*

# Q-Learning
## off-policy control

- Evaluate one policy while following another
- Can re-use experience gathered from old policies



**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., $\varepsilon$-greedy)
    Take action $A$, observe $R$, $S'$
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until $S$ is terminal

*Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.*

# Epsilon-greedy policy

Why should we follow an $\epsilon$-greedy policy? Isn't this suboptimal?

# Exercise Sheet 4

Fraunhofer
IIS

Fraunhofer-Institut für Integrierte Schaltungen IIS

# Thank you for your attention!