Reinforcement Learning

—

Exercise 5: Model-free Control

Nico Meyer

Fraunhofer IIS

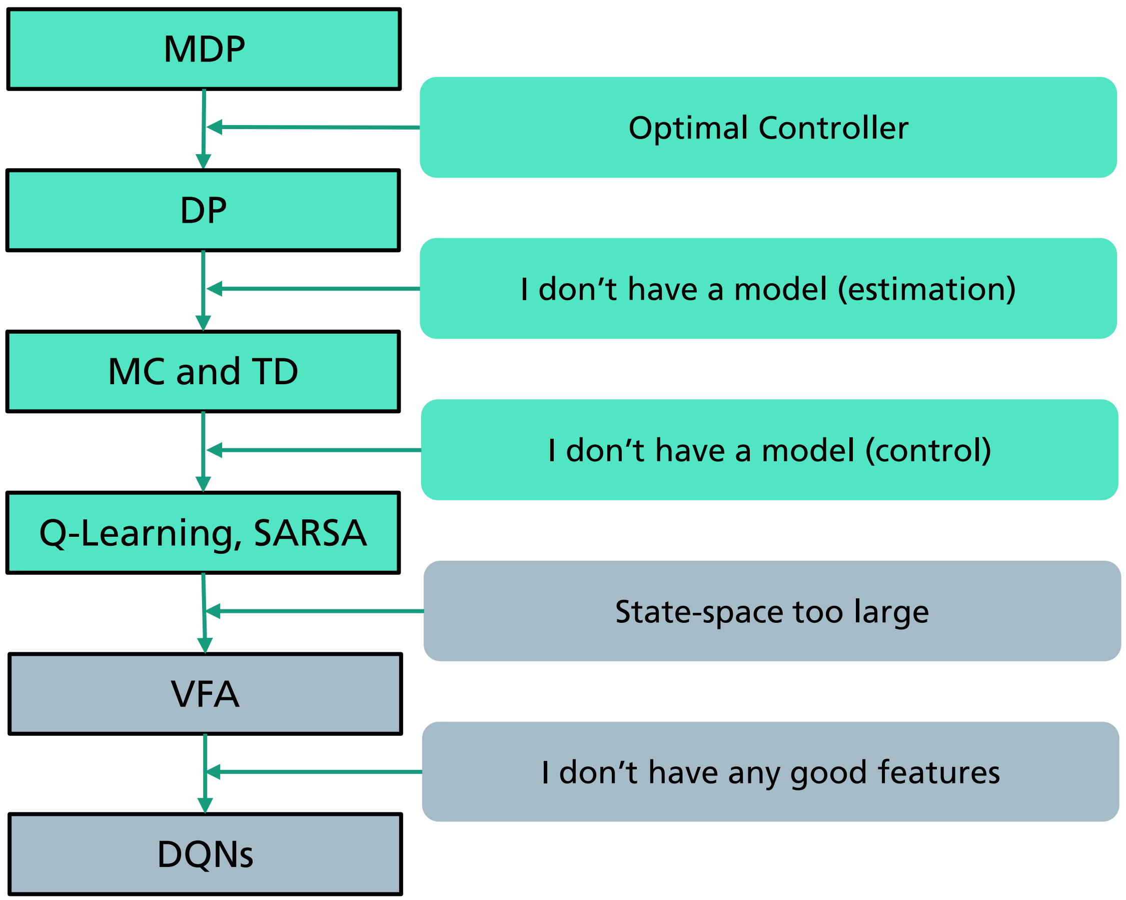
Fraunhofer-Institut für Integrierte Schaltungen IIS

Overview

Exercise Content

Week	*Date*	*Topic*	*Material*	*Who?*
0			*no exercises*	
1	23.04.	MDPs		Nico
2	30.04.	Dynamic Programming		Alex
3	07.05.	OpenAI Gym, PyTorch-Intro		Alex
4	14.05.	TD-Learning		Nico
5	22.05.	Practical Session (zoom@home)	**Attention: Lecture Slot!**	Nico + Alex
6	28.05.	TD-Control		Nico
7	04.06.	DQN		Nico
8	11.06.	VPG		Alex
9	18.06.	A2C		Nico
10	25.06.	Multi-armed Bandits		Alex
11	02.07.	RND/ICM		Alex
12	09.07.	MCTS		Alex
13	16.07.	BCQ		Nico

Overview
Overall Picture



Model-free Control

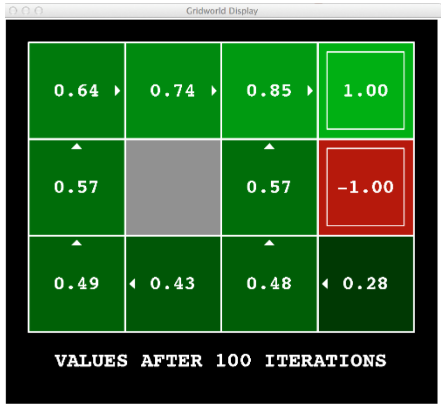
TD Methods

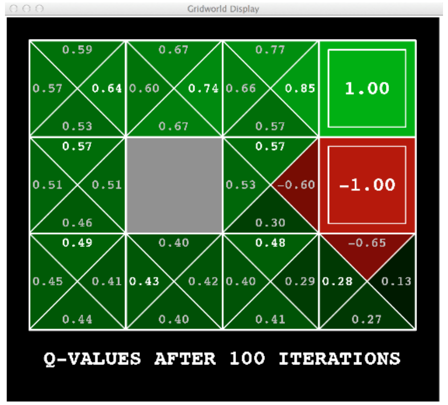
Recap

State-action-value function

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

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





Greedy Policy Improvement over Q:

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

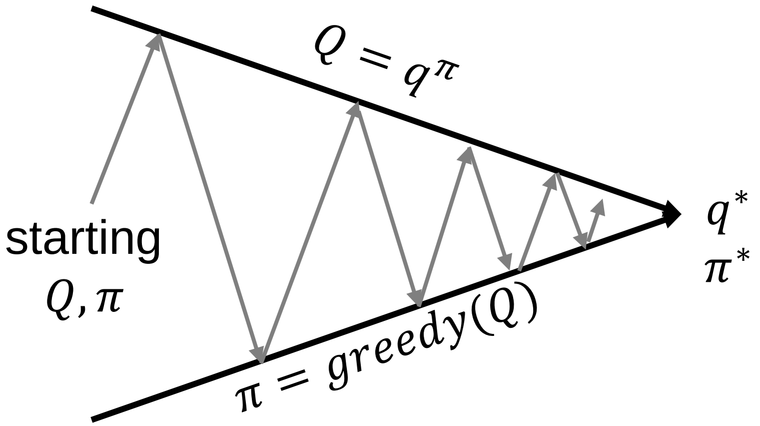
$$\forall s \in \mathcal{S}, \qquad Q^{\pi'}\big(s, \pi'(s)\big) \geq Q^{\pi}\big(s, \pi(s)\big)$$

Fraunhofer IIS

Recap

Model-free Control

- The (model-free) control problem:
  - **Given** experience samples $s(s, a, r, s')$
  - **Learn** a close-to optimal policy π
- Simple idea:
  - If we have calculated the value function for a given policy π (e.g., from MC/TD policy evaluation from last week), we can use it for deriving a better policy π' through greedy policy improvement over $Q(s)$



Policy Evaluation: Estimate $Q = q_{\pi}$
e.g., Monte Carlo Policy Evaluation

Policy Improvement: Generate $\pi' \geq \pi$
e.g., Greedy Policy Improvement over Q

Recap

Q-Learning and SARSA Algorithms

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 $s(s, a, r, s')$

In Exercise 4 we did:

Model-free Prediction: Evaluate the future, given the policy π.
(estimate the value function)

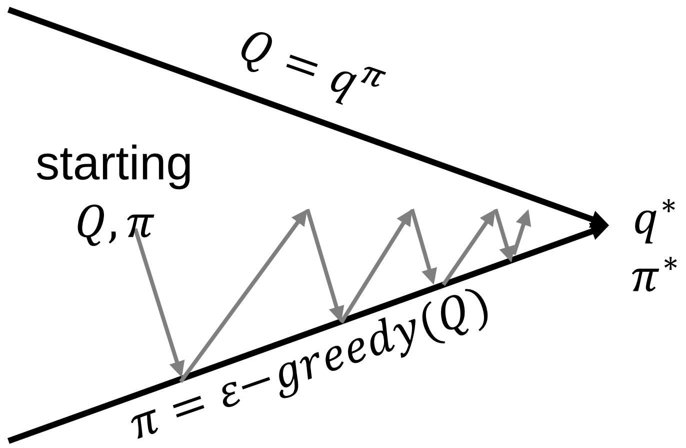
In Exercise 5 we will do:

Model-free Control: Optimize the future by finding the best policy π.
(optimize the value function)

Update every time step:

Policy Evaluation: Estimate $Q \approx q_\pi$
e.g., SARSA, Q-learning

Policy Improvement: Generate $\pi' \geq \pi$
e.g., ϵ-greedy Policy Improvement over Q

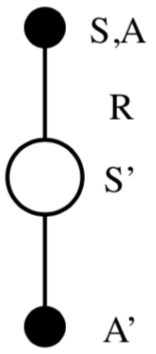


Fraunhofer IIS

Recap

SARSA: On-policy control

- Apply TD to $Q(s, a)$
- Use ε-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 $Q(terminal, \cdot) = 0$

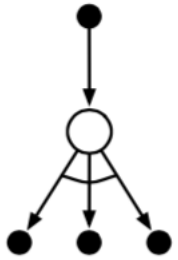
Loop for each episode:
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') -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

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

Recap

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., ε-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

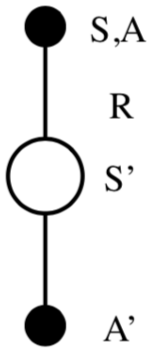
Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

Recap

Q-Learning vs. SARSA

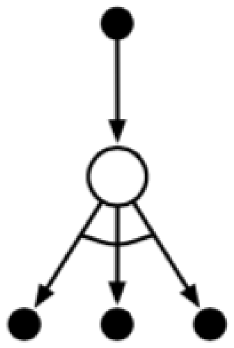
SARSA algorithm (on-policy control)

- \+ **Processes each sample immediately**
- \+ **Minimal update cost per sample**
- \- Requires a huge number of samples
- \- Requires careful schedule for the learning rate
- \- Makes minimal use of each sample
- \- The ordering of samples influences the outcome
- \- Exhibits instabilities under approximate representations
- \- Poses constraints on sample collection (on-policy)
- \- Requires careful handling on the policy greediness



Q-Learning algorithm (off-policy control)

- \+ **Processes each sample immediately**
- \+ **Minimal update cost per sample**
- \+ **Poses no constraints on sample collection (off-policy)**
- \- Requires a huge number of samples
- \- Requires careful schedule for the learning rate
- \- Makes minimal use of each sample
- \- The ordering of samples influences the outcome
- \- Exhibits (even more) instabilities under approximate representations



Epsilon-greedy policy

Why should we follow an ϵ-greedy policy? Isn’t this suboptimal?

Exercise Sheet 5

Model-free Control

Fraunhofer
IIS

Fraunhofer-Institut für Integrierte Schaltungen IIS

Thank you for your attention!