Reinforcement Learning

Jens Kober

j.kober@tudelft.nl

Outline

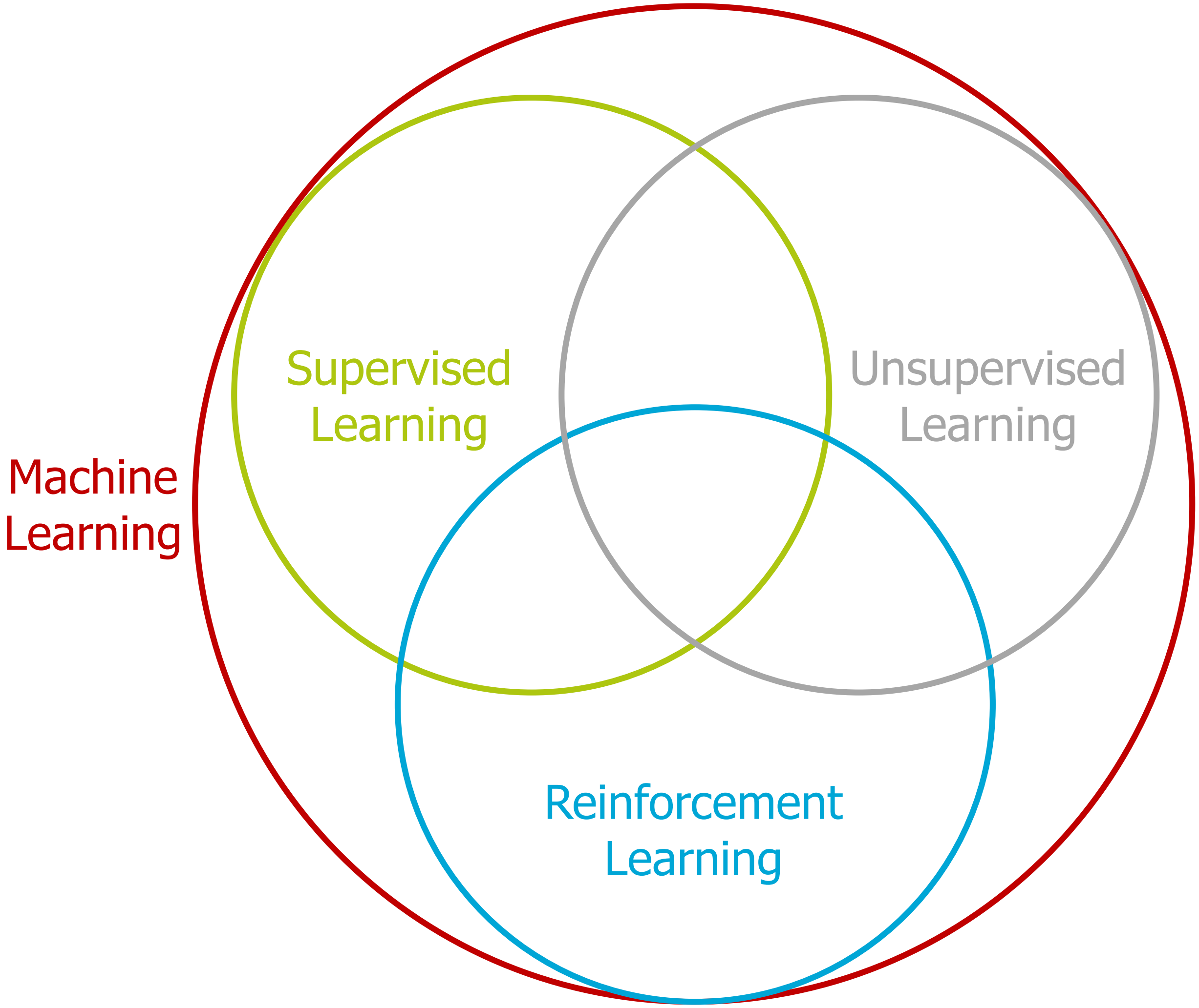
- Introduction
- Notation & basic concepts
- Types of RL algorithms
    - Optimal control
    - Value function methods
    - Policy search
- Summary & outlook

Types of Machine Learning

- Supervised learning
- Unsupervised learning
- Semi-supervised learning
- Reinforcement learning

- ➢ Definitions?
- ➢ Differences?
- ➢ Examples?

Types of Machine Learning



Reinforcement Learning

- Learning by trial & error

- Learning by rewards & punishments

Outline

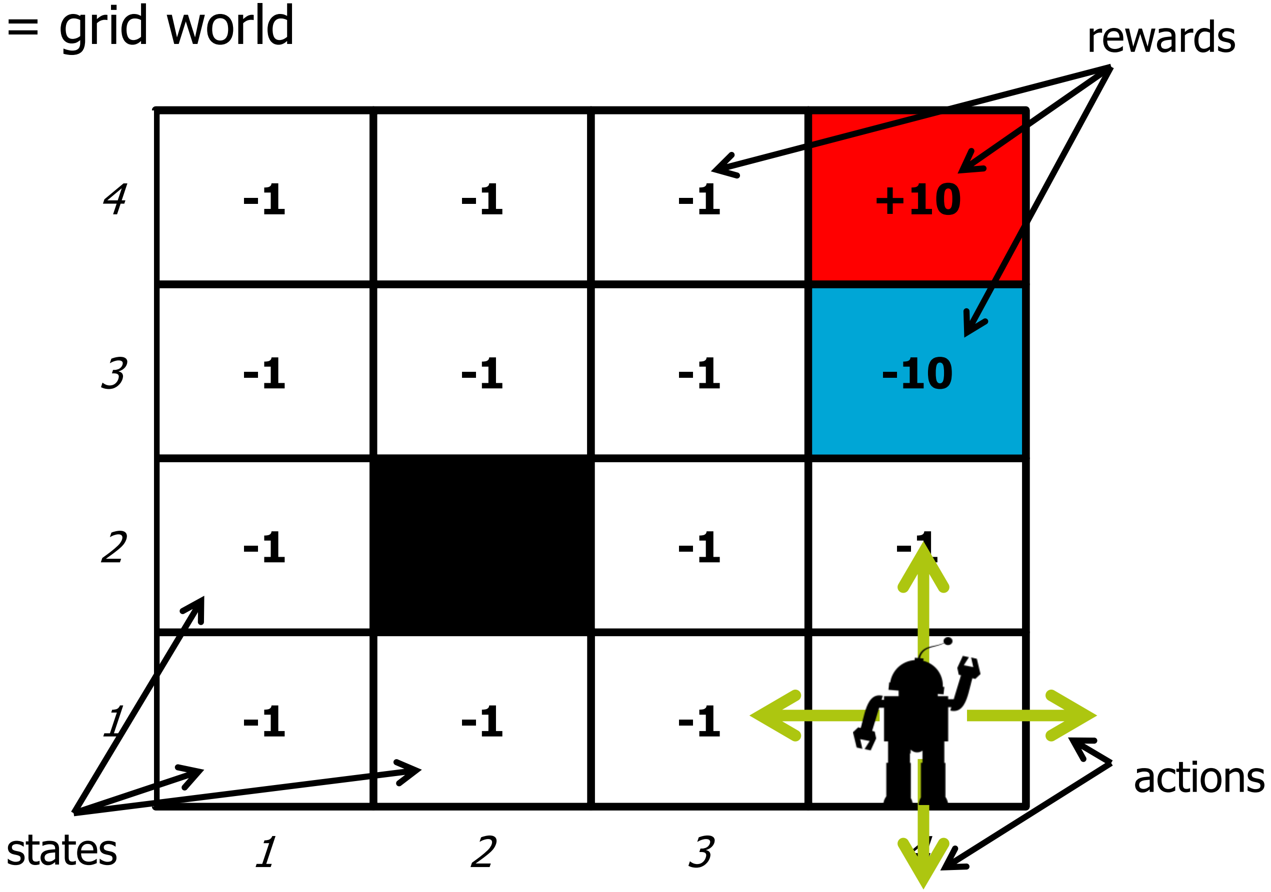
- Introduction
- Notation & basic concepts
- Types of RL algorithms
  - Optimal control
  - Value function methods
  - Policy search
- Summary & outlook

Classical RL Example

- Maze

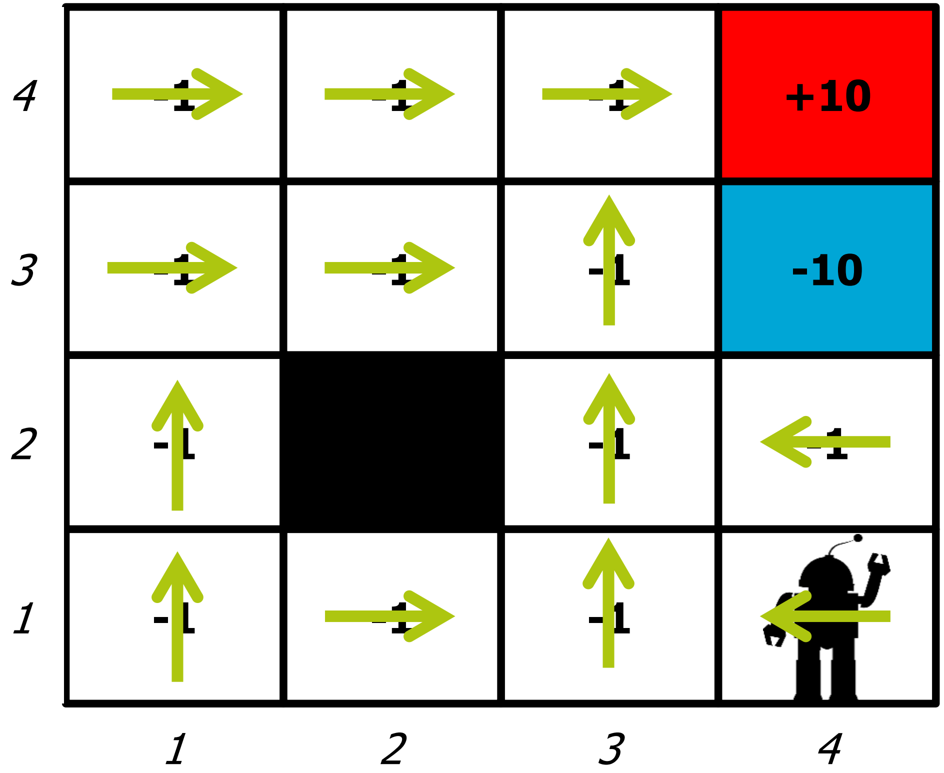
Classical RL Example

- Maze = grid world



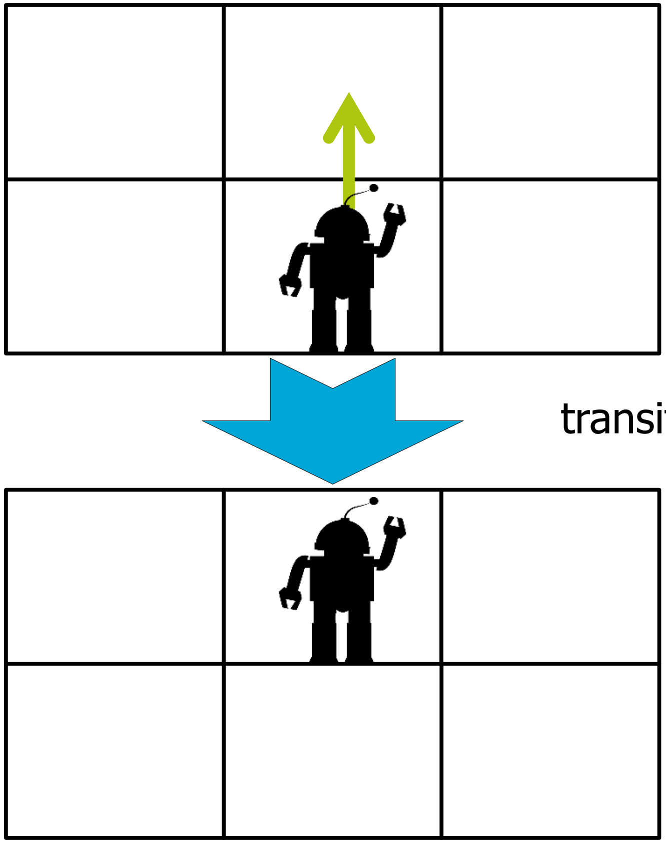
Classical RL Example

- Strategy = policy

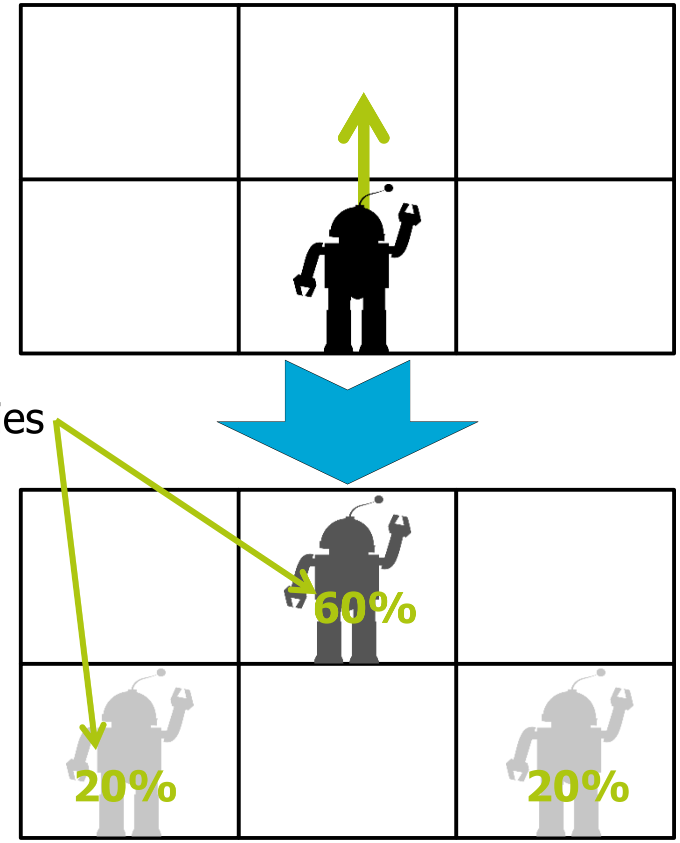


Classical RL Example

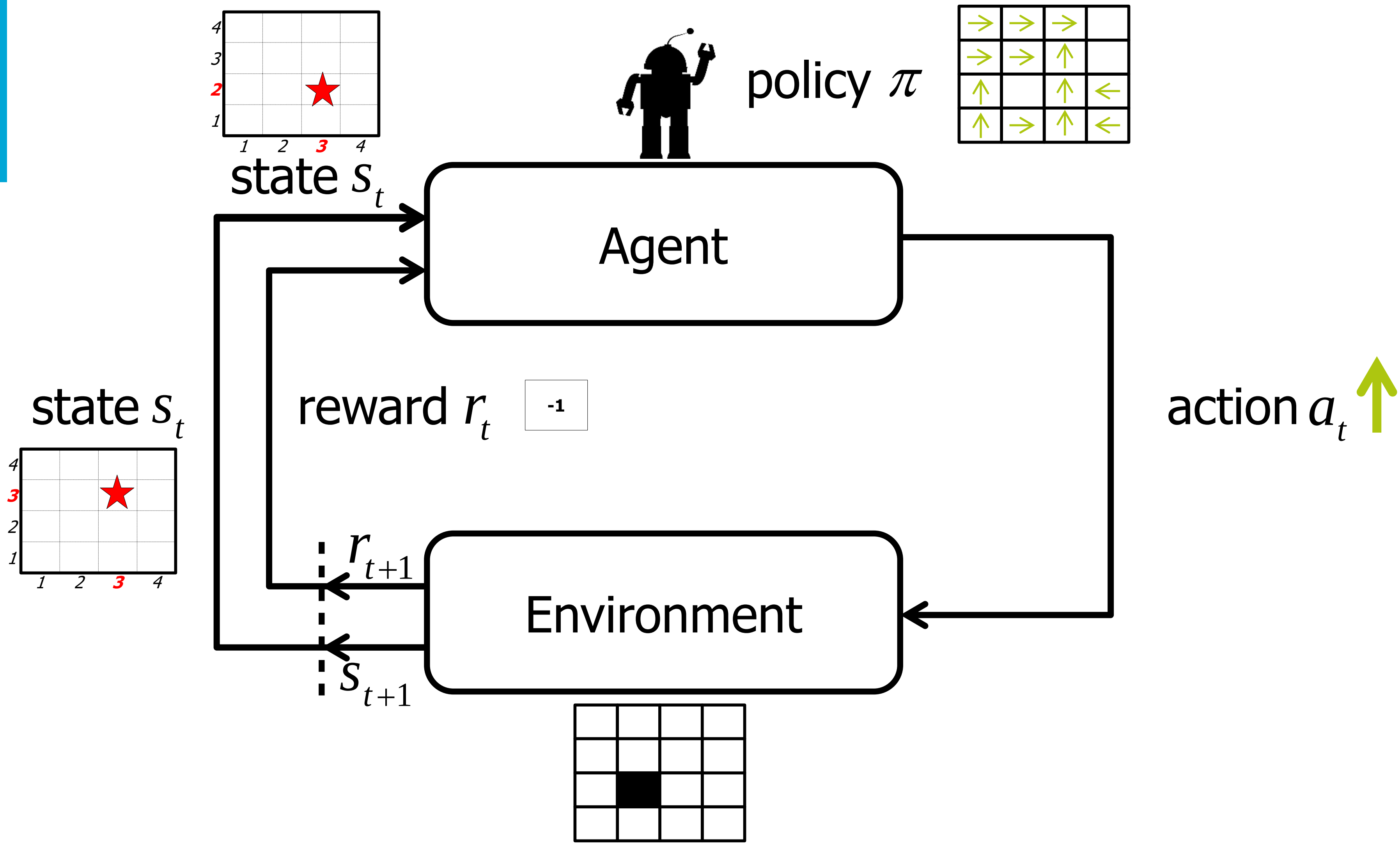
- Deterministic grid world



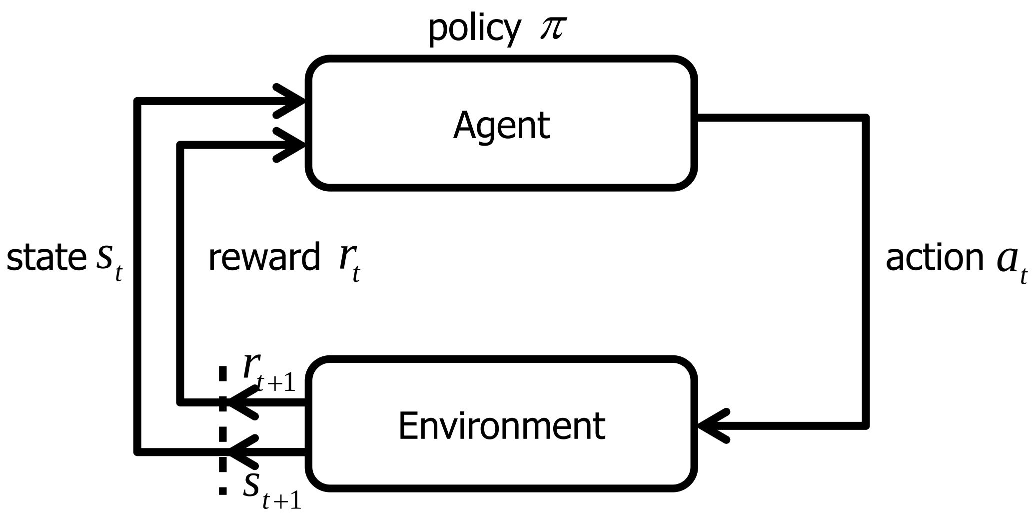
- Stochastic grid world



Formal Notation



Formal Notation



- $s \in \mathcal{S}$ set of states (discrete or continuous)
- $a \in \mathcal{A}$ set of actions (discrete or continuous)
- r reward $\mathcal{R}^{a}_{ss'} = E\left\{ r_{t+1} \mid a_t = a, s_t = s, s_{t+1} = s' \right\}$
- $\pi(s) = a$ policy

Goal: find π^* maximizing return R

Return R

- Finite Horizon H

$$R_t = r_{t+1} + r_{t+2} + \ldots + r_{t+H} = \sum_{h=1}^{H} r_{t+h}$$

- Discounted $0 \le \gamma < 1$

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+2} + \ldots = \sum_{h=0}^{\infty} \gamma^h r_{t+h+1}$$

- Average

$$R_t = \lim_{H \to \infty} \left(\frac{1}{H} \sum_{h=1}^{H} r_{t+h} \right)$$

Markov Property

- Transition probability

$$P\left(s_{t+1} \mid a_t, s_t, \cancel{a_{t-1}, s_{t-1}, \ldots, a_1, s_1}\right)$$

➢ Markov property

$$\mathcal{P}_{ss'}^{a} = P\left(s_{t+1} = s' \mid s_t = s, a_t = a\right)$$

- Markov decision process (MDP)

$$\mathcal{S}, \mathcal{A}, \mathcal{P}_{ss'}^{a}, \mathcal{R}_{ss'}^{a}, \gamma$$

Bellman Principle of Optimality

"An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."

Bellman, 1957

- Dynamic Programming
- Optimal Control

Value Functions

- State value function

$$V^{\pi}(s) = E^{\pi}\left\{\sum_{h=0}^{\infty}\gamma^{h} r_{t+h+1} \mid s_t = s\right\} = \sum_{s'\in S}\mathcal{P}_{ss'}^{\pi}\left[\mathcal{R}_{ss'}^{\pi} + \gamma V^{\pi}(s')\right]$$

$$V^{*}(s) = \max_{a\in\mathcal{A}}\left(\sum_{s'\in S}\mathcal{P}_{ss'}^{a}\left[\mathcal{R}_{ss'}^{a} + \gamma V^{*}(s')\right]\right)$$

- State-action value function

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

$$Q^{*}(s,a) = \sum_{s'\in S}\mathcal{P}_{ss'}^{a}\left[\mathcal{R}_{ss'}^{a} + \gamma V^{*}(s')\right]$$

$$= \sum_{s'\in S}\mathcal{P}_{ss'}^{a}\left[\mathcal{R}_{ss'}^{a} + \gamma\left(\max_{a'\in\mathcal{A}} Q^{*}(s',a')\right)\right]$$

Optimal Policy

- State-action value function

$$Q^*(s,a) = \sum_{s' \in S} \mathcal{P}_{ss'}^{a} \left[\mathcal{R}_{ss'}^{a} + \gamma \left(\max_{a' \in \mathcal{A}} Q^*(s',a') \right) \right]$$

➢ $\pi^*(s) = \arg\max_{a \in \mathcal{A}} \left(Q^*(s,a) \right)$

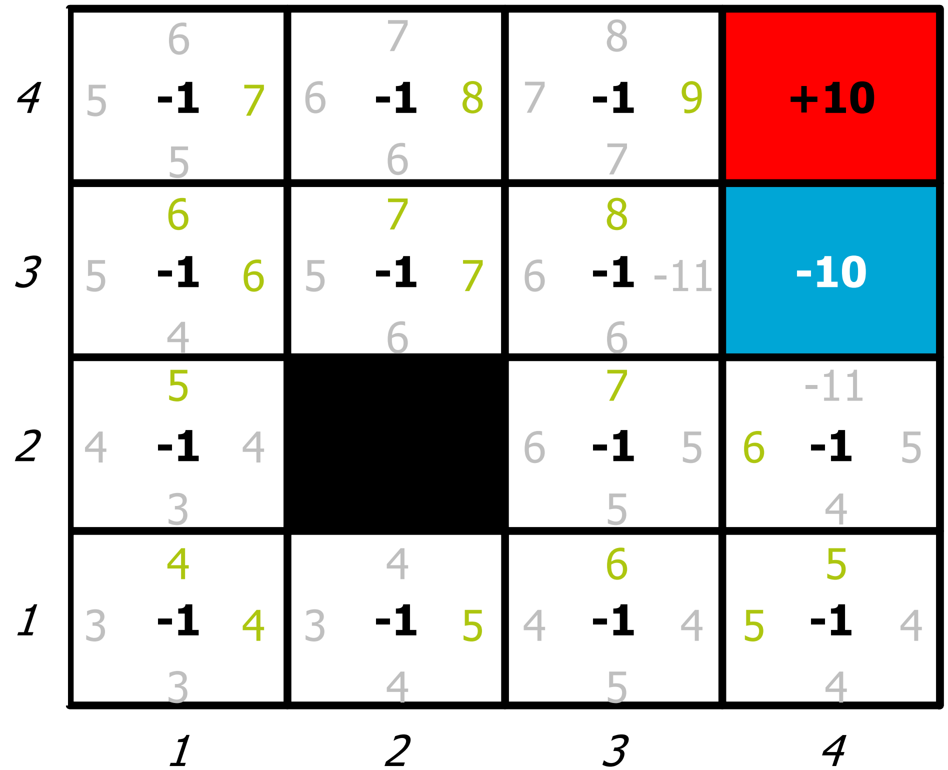
- State value function

$$V^*(s) = \max_{a \in \mathcal{A}} \left(\sum_{s' \in S} \mathcal{P}_{ss'}^{a} \left[\mathcal{R}_{ss'}^{a} + \gamma V^*(s') \right] \right)$$

➢ $\pi^*(s) = \arg\max_{a \in \mathcal{A}} \left(\sum_{s' \in S} \mathcal{P}_{ss'}^{a} \left[\mathcal{R}_{ss'}^{a} + \gamma V^*(s') \right] \right)$

Optimal State-Action Value Function

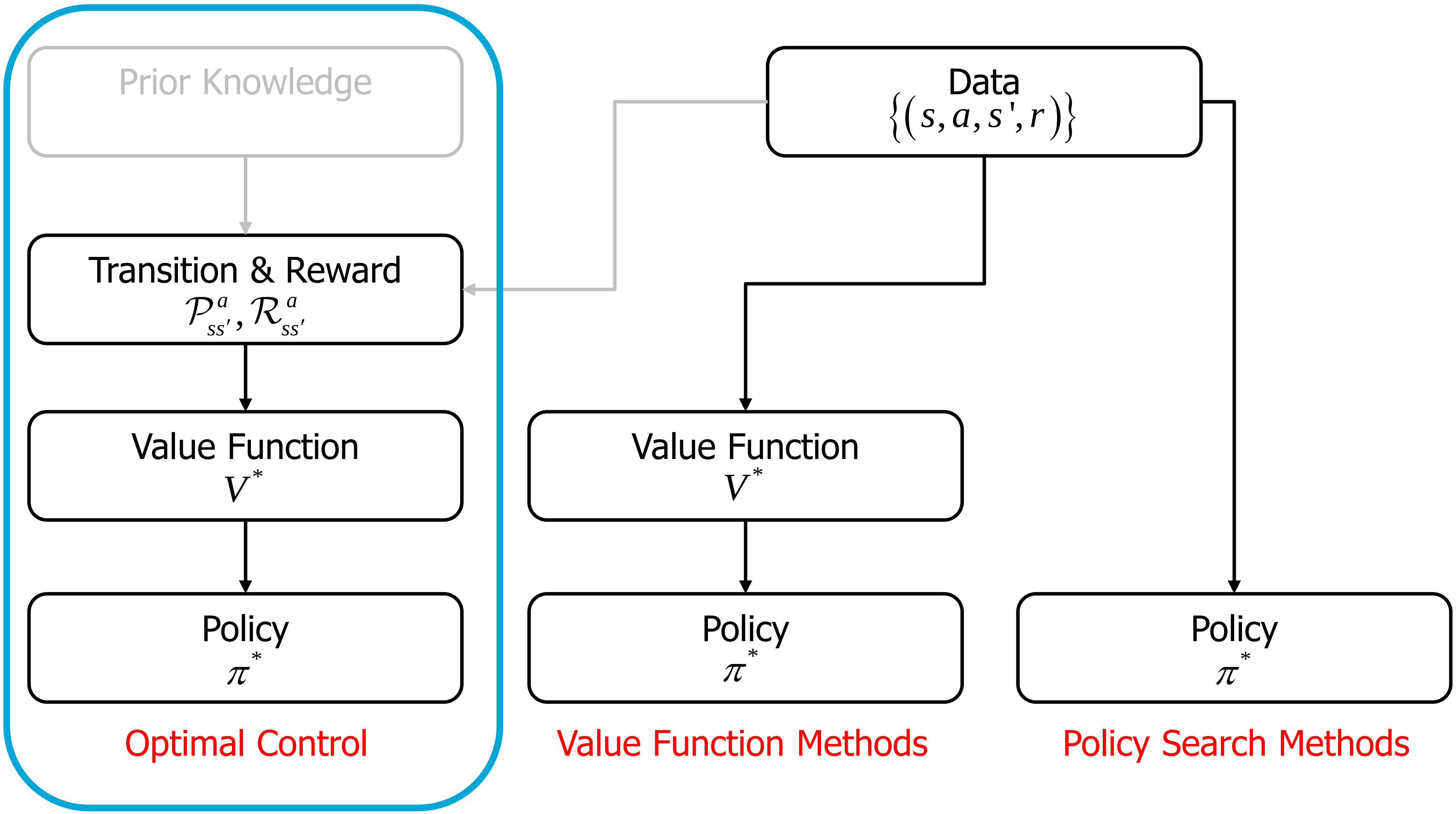
- max. 10 steps, stop at fields ± 10, stay at walls, deterministic, $\gamma = 1$, reward: -1 per step, ± 10 for reaching states



Outline

- Introduction
- Notation & basic concepts
- Types of RL algorithms
  - Optimal control
  - Value function methods
  - Policy search
- Summary & outlook

Types of RL Algorithms



Optimal Control

- Model-based: requires models of reward and transition functions
- Fast (no real world interactions required)
- Models can be also be learned
- RL often takes advantage of model errors
- Examples:
  - Policy Iteration
  - Value Iteration

Value Iteration

- Bellman optimality equation

$$Q^*(s,a) = \sum_{s' \in S} \mathcal{P}_{ss'}^a \left[\mathcal{R}_{ss'}^a + \gamma \left(\max_{a' \in \mathcal{A}} Q^*(s',a') \right) \right]$$

➢ Turn into an iterative update

Q-Iteration

repeat at each iteration i

  for all s, a **do**

$$Q_{i+1}(s,a) \leftarrow \sum_{s' \in S} \mathcal{P}_{ss'}^a \left[\mathcal{R}_{ss'}^a + \gamma \left(\max_{a' \in \mathcal{A}} Q_i(s',a') \right) \right]$$

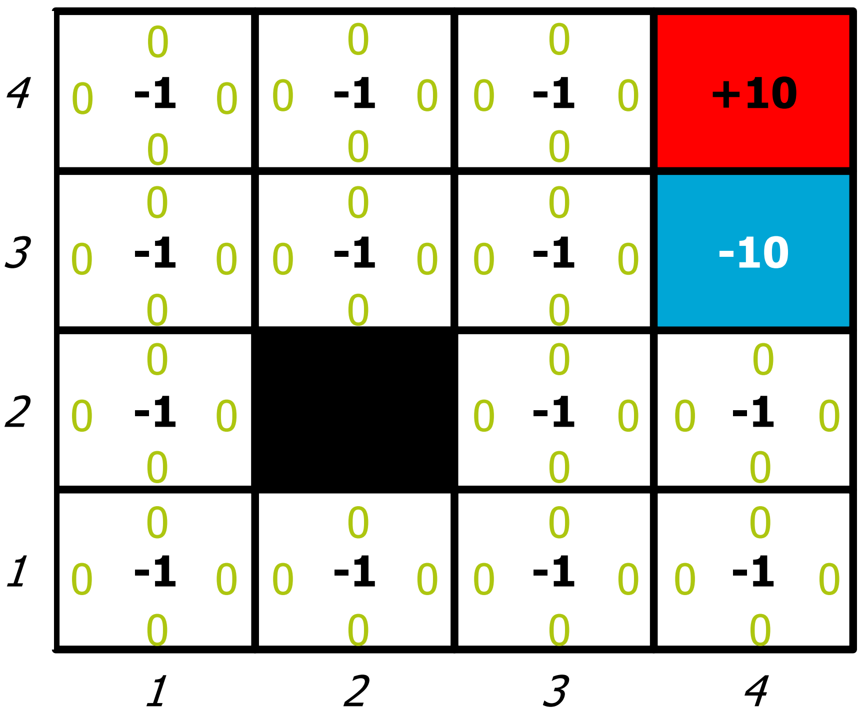
  end for

until convergence to Q^*

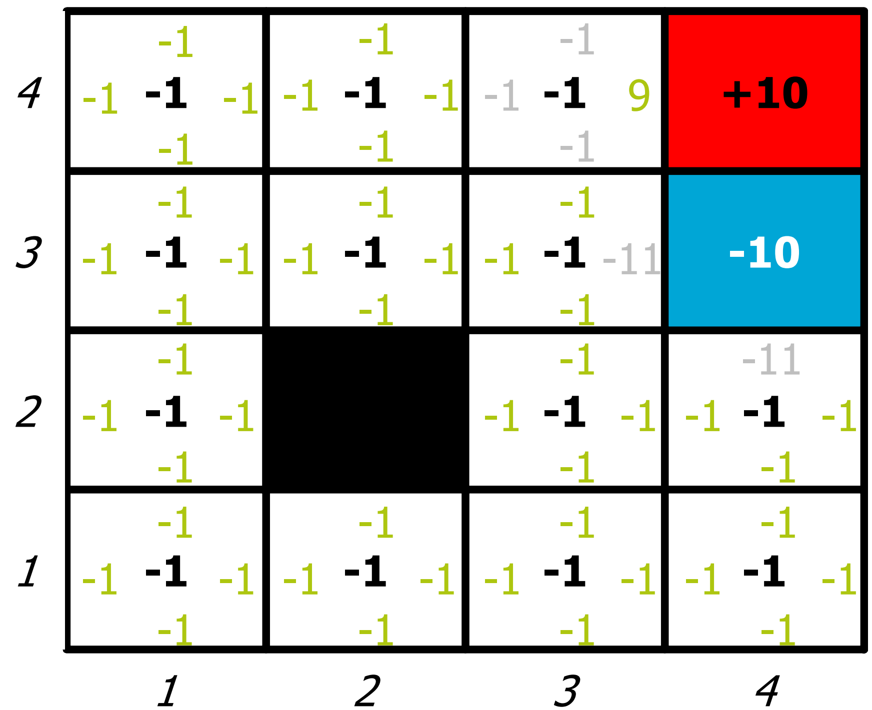
➢ Once Q^* available $\pi^*(s) = \arg\max_{a \in \mathcal{A}} \left(Q^*(s,a) \right)$

1st Iteration

Q_0

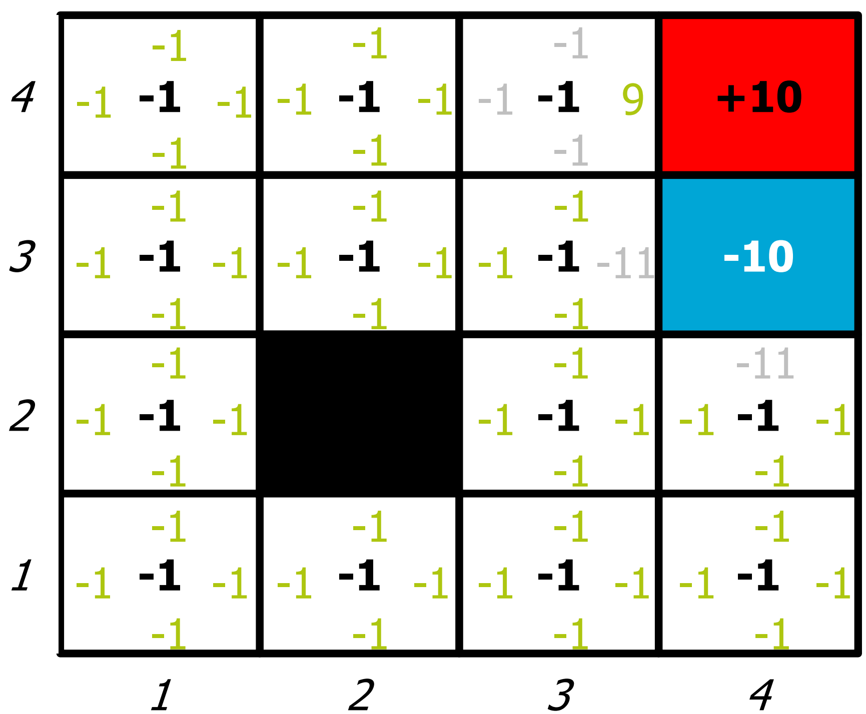


Q_1

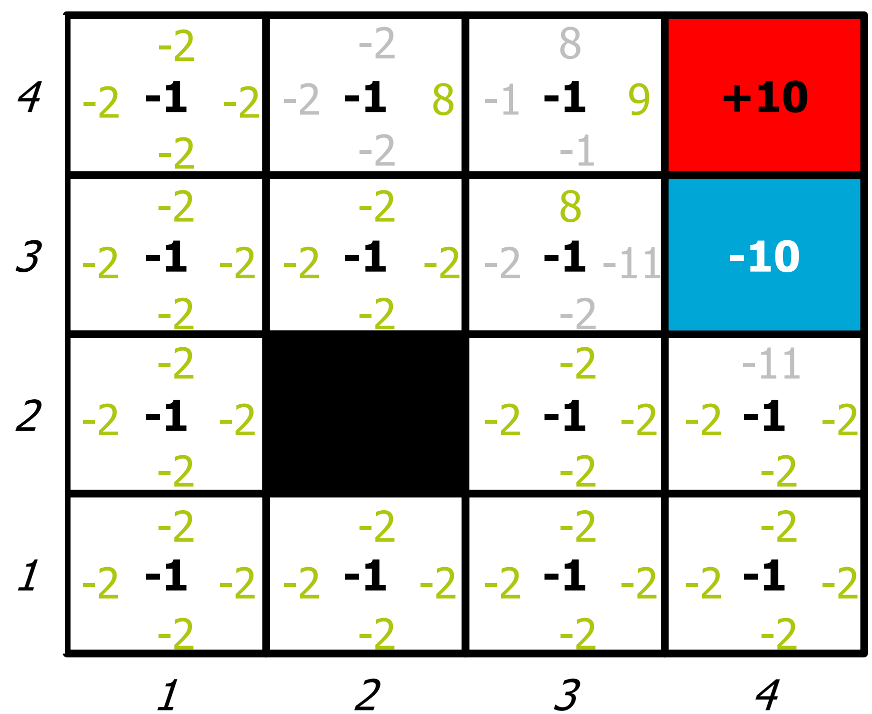


2^{nd} Iteration

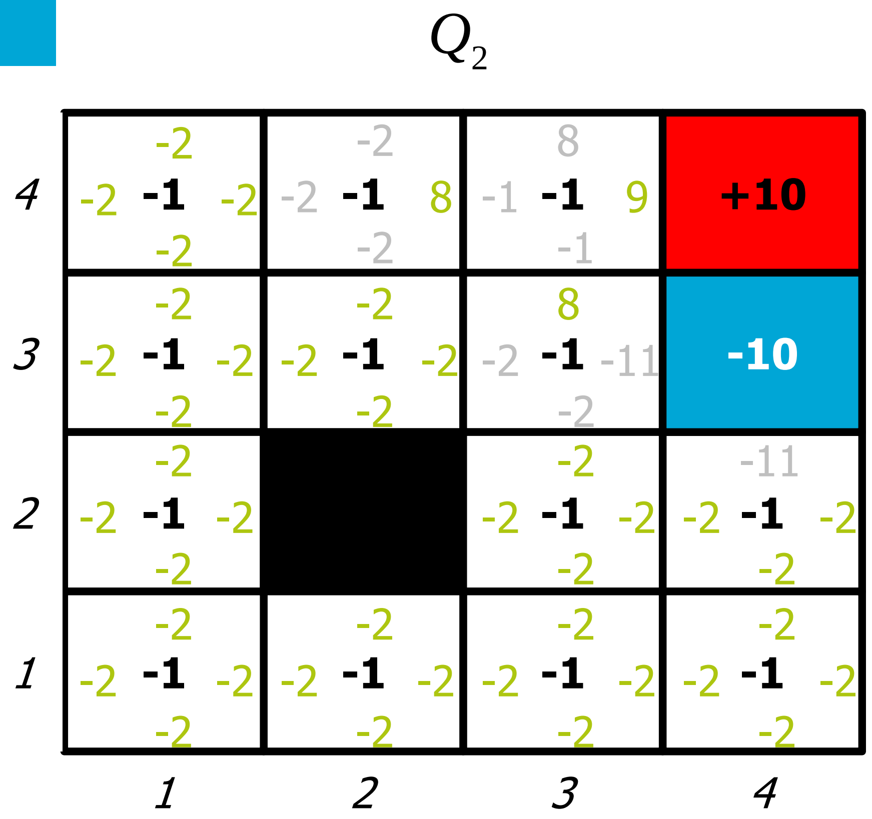
Q_1

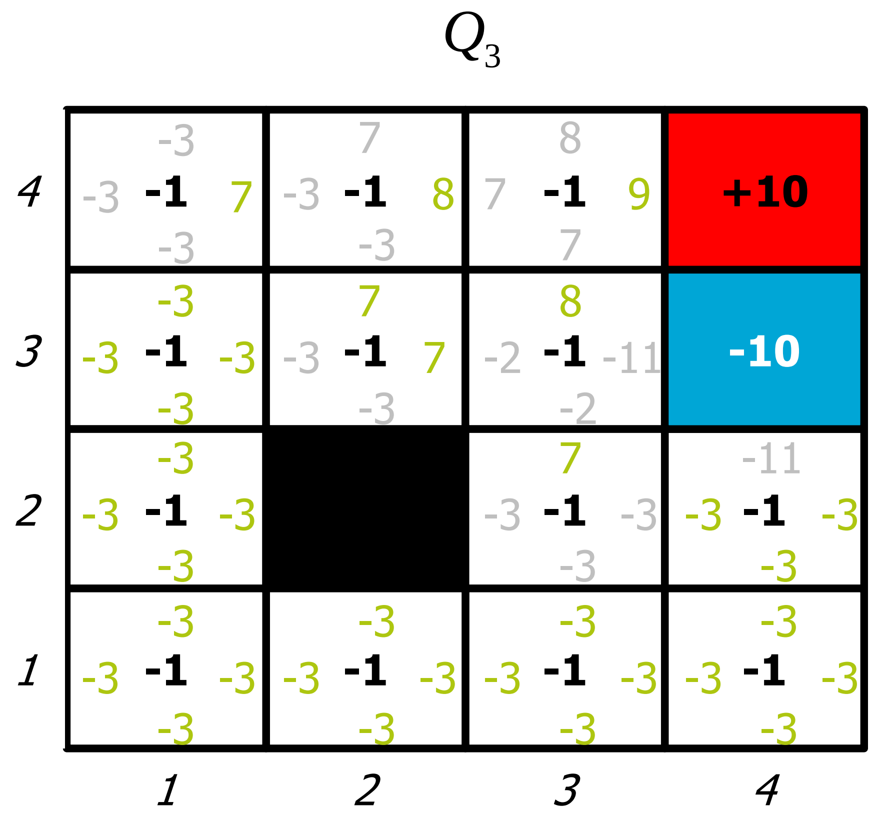


Q_2



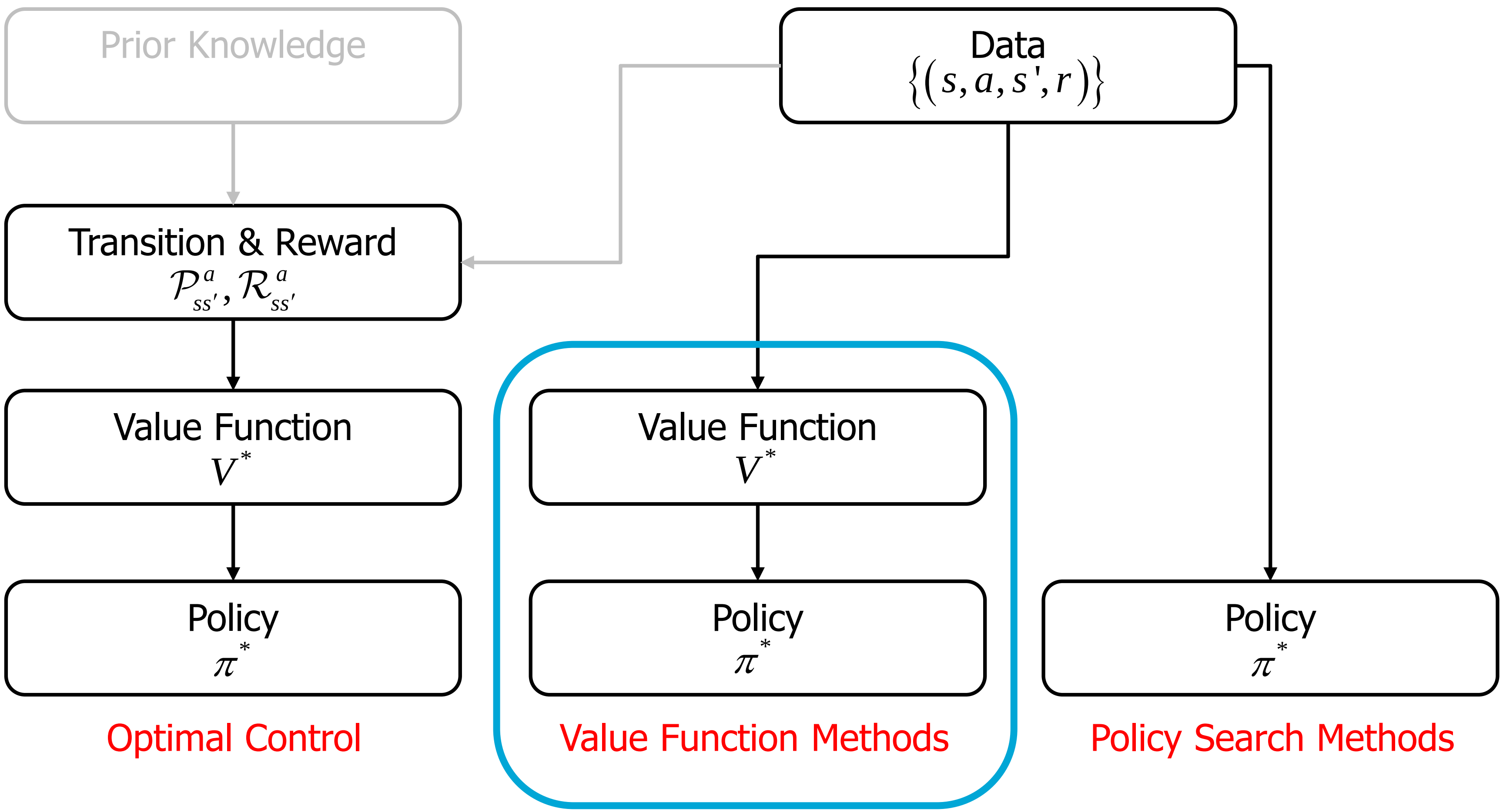
3rd Iteration





https://youtu.be/VCdxqn0fcnE

Types of RL Algorithms



Value Function Methods

- Model-free: (implicitly) learns models of reward and transition functions
- Global optimum (if everything is explored)
- Policy has global dependence
- Examples:
  - Monte Carlo
  - TD(λ)
  - Q-learning
  - SARSA

Temporal Difference Learning

- Use sample based estimate to update the value function

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha\left[r + \gamma \max_{a'\in\mathcal{A}} Q(s',a')\right]$$

learning rate

- Equivalently

$$Q(s,a) \leftarrow Q(s,a) + \alpha\underbrace{\left[r + \gamma \max_{a'\in\mathcal{A}} Q(s',a') - Q(s,a)\right]}_{\text{temporal difference error}}$$

Exploration

- Which action a to pick?
- Assume Q is optimal (greedy action)

$$a \leftarrow \arg\max_{a \in \mathcal{A}} \left(Q^{*}\left(s, a\right) \right)$$

- But Q is not optimal (yet)!

➢ We need to explore

ε-greedy policy:

- With probability ε pick a random action (exploration)
  $a \leftarrow \mathrm{rand}\left(\mathcal{A}\right)$
- With probability $\left(1-\varepsilon\right)$ pick the greedy action (exploitation)

$$a \leftarrow \arg\max_{a \in \mathcal{A}} \left(Q^{*}\left(s, a\right) \right)$$

Q-Learning

Q-Learning

loop

observe state s

$$a \leftarrow \arg\max_{a \in \mathcal{A}} \big(Q(s,a) \big)$$

with probability $\varepsilon, a \leftarrow \text{rand}(\mathcal{A})$

apply a, observe r and s'

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a' \in \mathcal{A}} Q(s',a') - Q(s,a) \right]$$

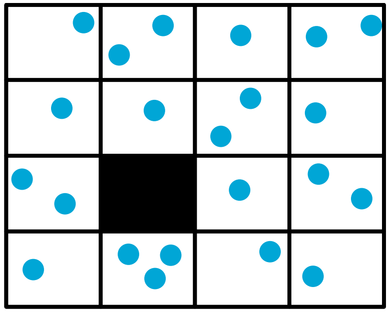
until convergence

Properties of Discussed Methods

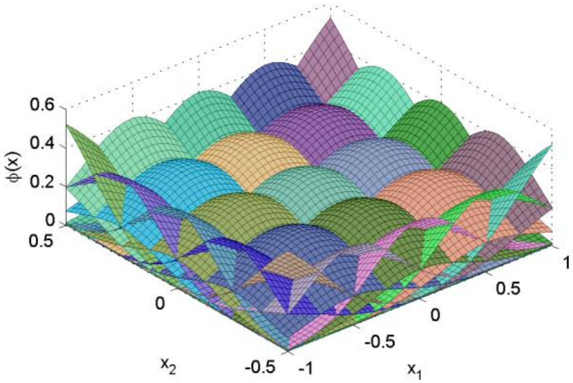
- Advantages
  - Generic framework
  - Very few assumptions
  - Guaranteed to converge to optimum
- Disadvantages
  - Can take lot of iterations
  - Infeasible for continuous states/actions

Dealing with Continuous States/Actions

- Discretization

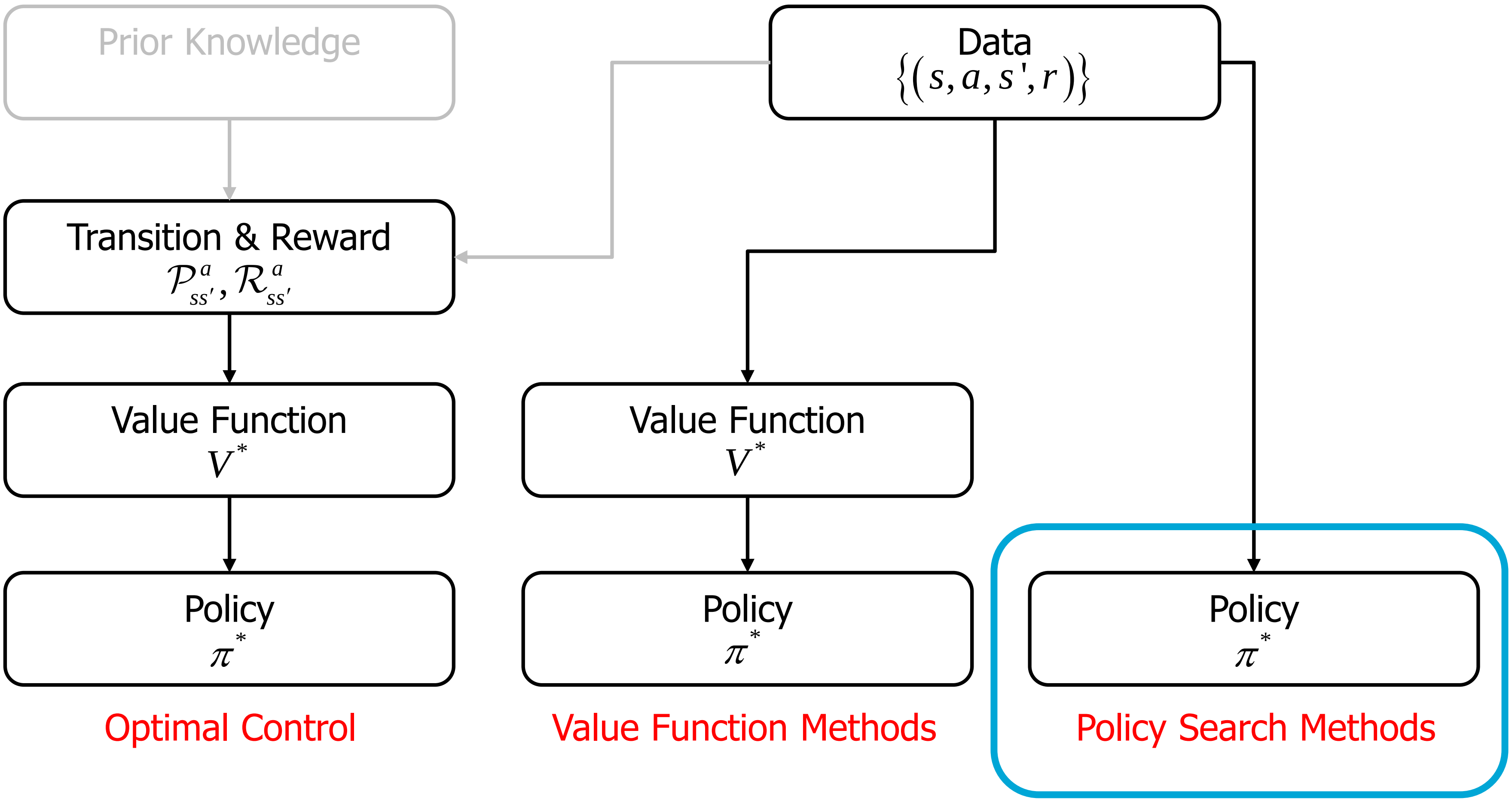


- Function approximation
  - Value function
  - Policy



https://youtu.be/nM1HTp_P3lY

Types of RL Algorithms



Policy Search Methods

- Model-free
- Local optimum
- Usually parametrized policies

- Examples:
  - Gradient-based
  - Expectation-maximization inspired
  - General optimization

https://youtu.be/qtqubguikMk

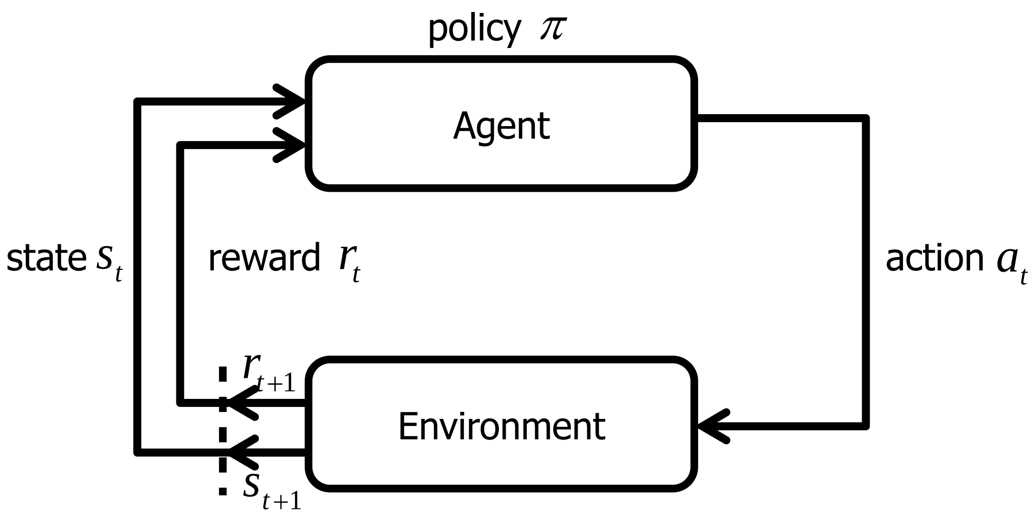
Outline

- Introduction
- Notation & basic concepts
- Types of RL algorithms
  - Optimal control
  - Value function methods
  - Policy search
- Summary & outlook

Summary

Reinforcement learning

- Inspired by human & animal learning
- Reward as feedback signal
- Model-free, adaptive optimal control



Goal: find π^* maximizing return R

Challenges of (Robot) RL

- Curse of dimensionality
- Exploration-exploitation trade-off
- Real-world samples
- Model uncertainty
- Goal specification

Tractability Through:

- Representations
  - State and/or action discretization
  - Value function approximation
  - Pre-structured policies
- Prior knowledge
  - Demonstrations
  - Task Structure
- Models
  - Mental rehearsal