Model Predictive Control and Reinforcement Learning
– Introduction (RL part) –

Joschka Boedecker and Moritz Diehl

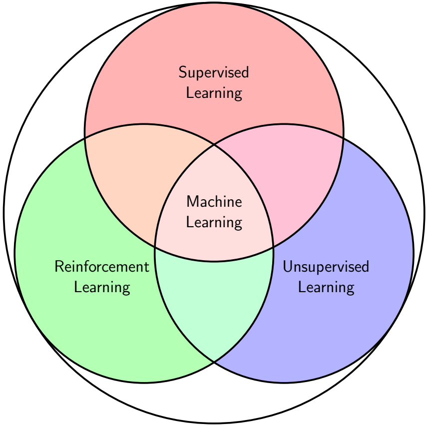
University Freiburg

July 26, 2021

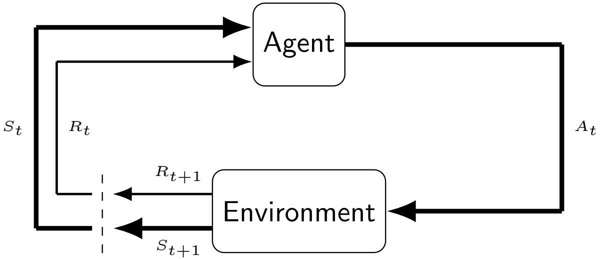
Acknowledgement

Slide contents are partially based on *Reinforcement Learning: An Introduction* by Sutton and Barto and the Reinforcement Learning lecture by David Silver.

Reinforcement Learning in Machine Learning



Agent and Environment



Time steps t: $0, 1, 2, \ldots$
States: $S_0, S_1, S_2, \ldots$
Actions: $A_0, A_1, A_2, \ldots$
Rewards: $R_1, R_2, R_3, \ldots$

Rewards

- A reward R_t in time step t is a **scalar** feedback signal.
- R_t indicates how well an agent is performing **at single time step** t.
- The agent aims at maximizing the expected discounted **cumulative reward** $G_t = R_{t+1} + \gamma^1 R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-(t+1)} R_T$.
  T can be inifinite.

Reward Hypothesis

All of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward).

Examples:

- Chess: +1 for winning, -1 for losing
- Walking: +1 for every time step not falling over
- Investment Portfolio: difference in value between two time steps

Exploration and Exploitation

- Fundamental problem in Reinforcement Learning
- The agent has to exploit what it knows in order to obtain high reward (**Exploitation**)...
- ...but it has to explore to possibly do better in the future (**Exploration**).

Example: You want to go out for dinner. Do you...

- go to your favourite restaurant
- or try a new one?

Markov Decision Processes

A finite Markov Decision Process (MDP) is a 4-tuple $\langle \mathcal{S}, \mathcal{A}, p, \mathcal{R} \rangle$, where

- $\mathcal{S}$ is a finite number of states,
- $\mathcal{A}$ is a finite number of actions,
- p is the transition probability function $p : \mathcal{S} \times \mathcal{R} \times \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$,
- and $\mathcal{R}$ is a finite set of scalar rewards. We can then define expected reward $r(s, a) = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$ and $r(s, a, s') = \mathbb{E}[R_{t+1} | S_t = s, A_t = a, S_{t+1} = s']$.

Markov Property

A state-reward pair (S_{t+1}, R_{t+1}) has the Markov property iff:

$$\mathrm{Pr}\{S_{t+1}, R_{t+1} | S_t, A_t\} = \mathrm{Pr}\{S_{t+1}, R_{t+1} | S_t, A_t, \ldots, S_0, A_0\}.$$

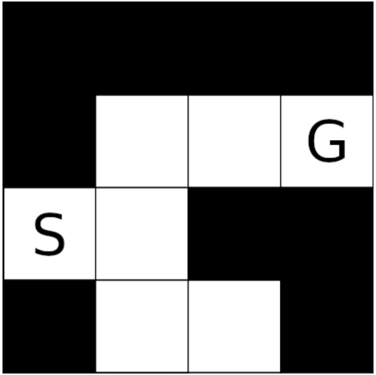
The future is independent of the past given the present.

Components of RL Systems

- Policy: defines the behaviour of the agent
  - is a mapping from a state to an action
  - can be stochastic: $\pi(a|s) = \mathbb{P}[A_t = a|S_t = s]$
  - or deterministic: $\pi(s) = a$
- Value-function: defines the expected value of a state or an action
  - $v_\pi(s) = \mathbb{E}[G_t|S_t = s]$ and $q_\pi(s,a) = \mathbb{E}[G_t|S_t = s, A_t = a]$
  - Can be used to evaluate states or to extract a good policy
- Model: defines the transitions between states in an environment
  - p yields the next state and reward
  - $p(s', r|s, a) = \text{Pr}\{S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a\}$

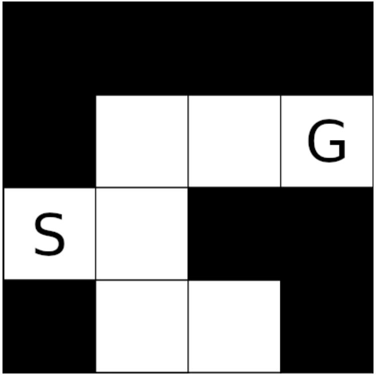
Maze Example: Policy

- Rewards: -1 per time step
- Actions: up, down, left, right
- States: location of the agent



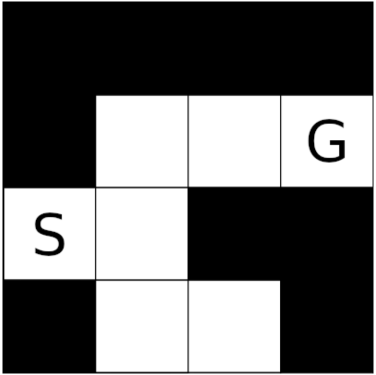
Maze Example: Value-function

- Rewards: -1 per time step
- Actions: up, down, left, right
- States: location of the agent

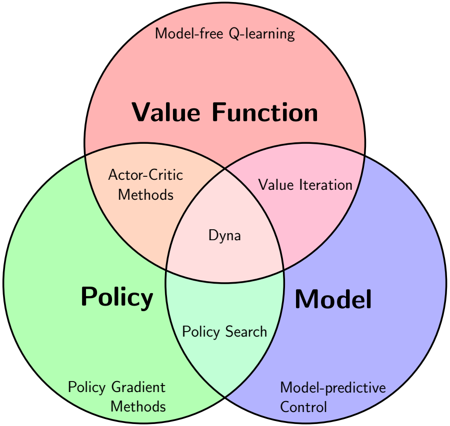


Maze Example: Model

- Rewards: -1 per time step
- Actions: up, down, left, right
- States: location of the agent

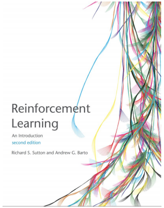


RL Overview



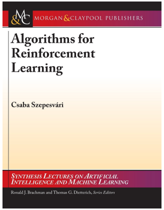
Model-free Q-learning
Value Function
Actor-Critic Methods
Value Iteration
Dyna
Policy
Model
Policy Search
Policy Gradient Methods
Model-predictive Control

Literature



Reinforcement Learning:
An Introduction (Sutton
and Barto, 2018) http://incompleteideas.net/book/the-book.html

Algorithms for Reinforcement Learning (Szepesvári, 2010)
https://sites.ualberta.ca/~szepesva/RLBook.html



Discussion

Where would you apply Reinforcement Learning?