Zurich University of Applied Sciences

Reinforcement Learning (RL)

Gabriel Eyyi, 29.11.17

www.zhaw.ch/datalab

Content

- Reinforcement Learning Setting
- **Reinforcement Learning: Model-Free Solution Method**
- OpenAI Gym & Demo

Goal

- Goal of this presentation:
  - to give a short introduction to Reinforcement Learning
  - to show you the basic elements of RL
  - to present a basic solution method and give you an intuition

- Source
  - See last slide

Introduction

- „Good and evil, reward and punishment, are the only motives to a rational creature: these are the spur and reins whereby all mankind are set on work, and guided.“
  –John Locke

Reinforcement Learning (RL)

- Reinforcement Learning (RL) is learning what to do —how to map situations to actions— so as to maximize a numerical reward signal.

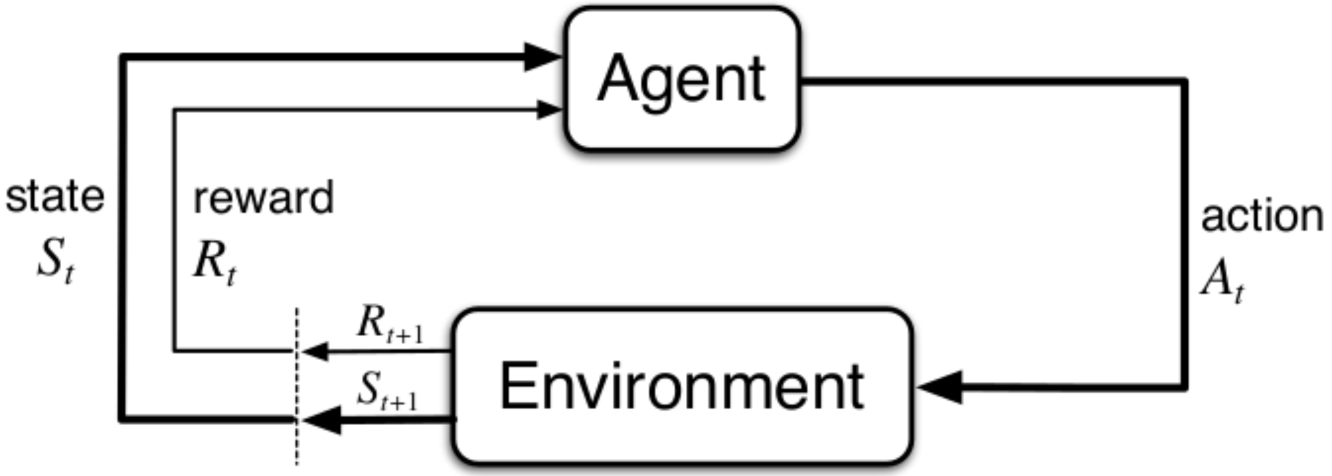
- Examples:
  - Learning to drive a car:
    - Environment's response affects our subsequent actions
    - We find out the effects of our actions later

Reinforcement Learning (RL)

- Reinforcement Learning is:
  - a problem,
  - a class of solution methods that work well on the problem,
  - and the field that studies this problem and its solutions methods.

- Reinforcement Learning Problem is:
  - More general and thus more difficult
  - Learning has to be based on considerably less knowledge

Reinforcement Learning Problem



- → Goal: Learning a mapping from states to actions in order to maximize a scalar reward (reinforcement signal)

Elements of Reinforcement Learning

- An MDP = $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R})$
  - State is a unique characterization of all that is important:
    - Chess: Configuration of board pieces of both black and white
  - Actions can be used to control the system state:
    - Chess: All possible moves
  - Transition function:
    - Markov property: $P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \ldots) = P(s_{t+1}|s_t, a_t) = \mathcal{T}(s, a, s_{t+1})$
  - Reward function:
    - The reward function specifies rewards for being in a state, or doing some action in a state
    - Specifies implicitly the goal of learning
    - Gives a direction in which way the system should be controlled

Reinforcement Learning vs. Supervised Learning

- Supervised Learning:
  - Instructive feedback
  - Objective → generalization

- Reinforcement Learning:
  - Evaluative Feedback
  - Learning from interaction
  - Objective → maximize a reward

Elements of Reinforcement Learning

- Policy:
  - Deterministic policy $\pi\colon \mathcal{S} \longrightarrow \mathcal{A}$
    - Selecting action $\pi(s)$ in state s
  - Stochastic policy $\pi\colon \mathcal{S} \times \mathcal{A} \longrightarrow [0, 1]$
    - Specifies a probability distribution over $\mathcal{A}$ for each state s, where $\pi(s, a)$ denotes the probability to choose action a in state s
  - The agent interacts with the MDP.
  - Control the environment
  - → function or lookup table

- Optimality Criteria
  - What is the actually being optimized?
  - What is the goal of the agent?
  - → Gathering reward

Elements of Reinforcement Learning

- State Value Function V:
  - Indicates what is good in the long run

$$V^{\pi}(s) = E\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k} \,\middle|\, s_t = s, \pi\right\}$$

- **State-Action Function Q:**

$$Q^{\pi}(s,a) = E\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k} \,\middle|\, s_t = s, a_t = a, \pi\right\}$$

- **Optimal Policy π*:**

$$\pi^*(s) = \arg\max_a Q(s, a)$$

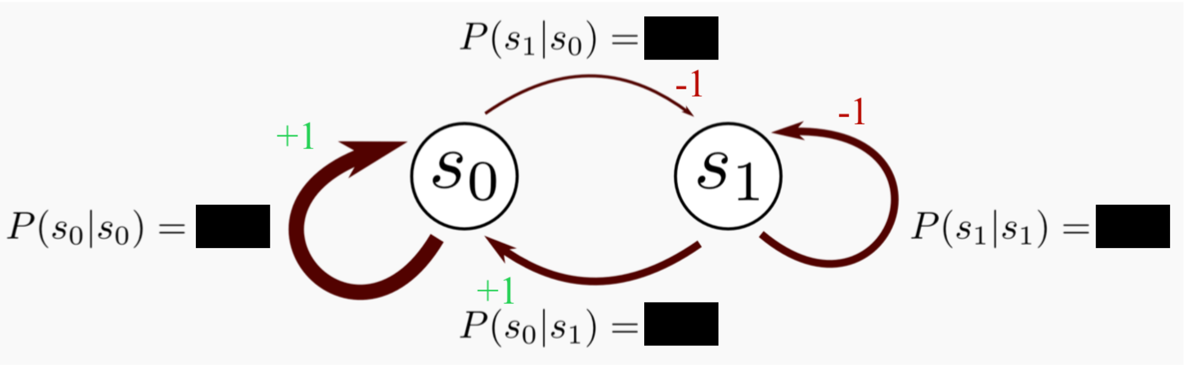
- Both function satisfy certain recursive properties:
  - → Bellman Equation
  - → The Bellman equation expresses a relationship between the **value of a state s** and the values of its **successor states s'.**

Main characteristics of RL

- Trial-and-error search
  - Opportunity for active exploration: Needs trade-off between exploration and exploitation
    - The agent must try a variety of actions and progressively favor those that appear to be beast.
- Delayed reward
  - Credit assignment problem
- Feedback
  - Evaluative feedback (not instructive as in supervised learning)
- Representations
  - What and how should be represented?

Reinforcement Learning

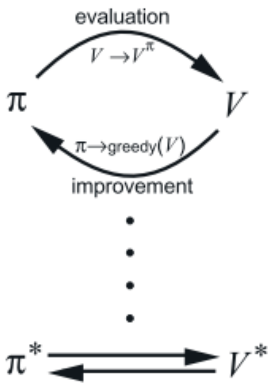
- Still assume a MDP
  - a set of states $s \in \mathcal{S}$
  - a set of actions (per state) $a \in \mathcal{A}$
  - a transition function $\mathcal{T}(s, a, s')$
  - a reward function $\mathcal{R}(s, a, s')$
- Still looking for a optimal policy

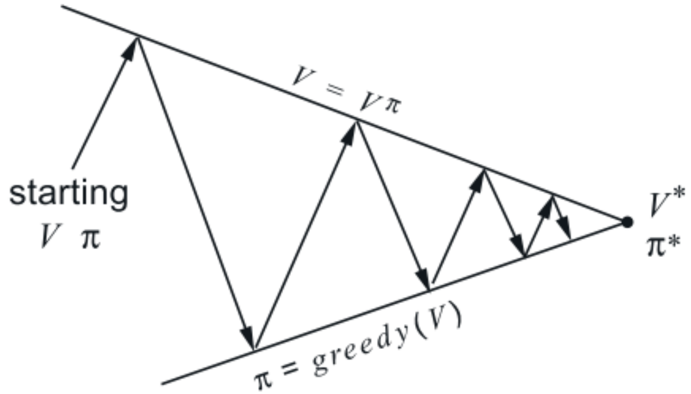


- Challenge:
  - $\mathcal{T}$ or $\mathcal{R}$ are unknown
    - I.e. we don't know which states are good or what the actions do
    - Must actually try actions and states out to learn

Generalized Policy Iteration (GPI)

- **The policy evaluation step:**
  - Estimate the value of the current policy π
  - Main purpose:
    - Gather information about the policy





- **The policy improvement step:**
  - Evaluate the values of the actions for every state
  - Computes an improved policy π'

Model-based vs Model-free

- Model-based (indirect RL):
  - Learn the transition and reward model from interaction with the environment
  - After that, when the model is (approximately or sufficiently) correct
    - → apply DP methods

- Model-free (direct RL):
  - Estimating the values for actions, without even estimating the model of the MDP.

Reinforcement Learning: Model-free

- RL is primarily concerned with how to obtain an optimal policy when such a model is not available.
- → This leads to :
    - The need for **sampling** and **exploration**
- The policy evaluation step:
    - simulate the policy and estimate its utility from the sampled execution traces
- Implicit representation of the policy
    - Policy is computed on-the-fly for each state based on the value function
    - Only value function function is stored

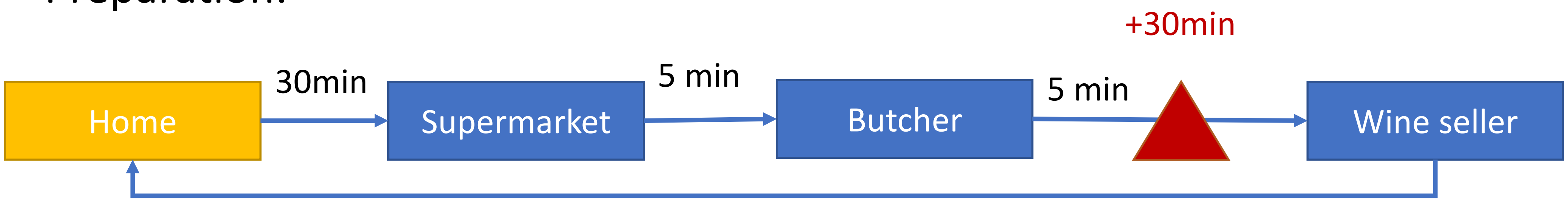
Temporal Difference (TD) Learning

- Temporal difference learning algorithms learn estimates of values based on other estimates.
  - Each step in the environment generates a learning example which can be used to bring some value in accordance to the immediate reward and the estimated value of the next state or **state-action pair.**

- MDP is often unknown
  - → to get information by interacting with the environment

Temporal Difference Learning: Example

- Organizing a dinner with friends:
  - Predict what time your guest can arrive?
  - Preparation:



1) 18:00h
2) 17:40h
3) 18:10h

The bottom line of this example is that you can adjust your estimate about what time your will be back home every time you have obtained **new information about in-between steps**.

Temporal Difference Learning

- **Main principle of TD Learning:**
  - Adjust your estimate every time you have obtained new information
    - Each time you can adjust your estimate based on actually experienced times of parts of your path.
  - Learn their value estimates based on estimates of other values
    →bootstrapping

- **Advantages**
  - No model of the MDP is required
  - No full sweeps through the full state space are needed

Zurich University
of Applied Sciences

Q-learning

- Basic idea:
  - Incrementally estimate Q-values for actions, based on feedback (i.e. rewards) and the agent‘s Q-value functions.

$$\underset{k+1}{Q}(s_t, a_t) \leftarrow \underbrace{\underset{k}{Q}(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a \underset{k}{Q}(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{\underset{k}{Q}(s_t, a_t)}_{\text{old value}} \right)$$

  - Converge to the optimal policy regardless of the exploration policy being followed
    - Assumption:
      - Each state-action pair is visited an infinite number of times
      - Learning parameter is decreased appropriately.
- Demo

www.zhaw.ch/datalab

Conclusion

- Reinforcement learning extends the domain of machine learning to a broad area of control and decision problems that cannot be tackled with supervised or unsupervised learning techniques.

OpenAI

- OpenAI is a non-profit AI research company, discovering and enacting the path to **safe artificial general intelligence (AGI).**
- OpenAI was founded in late 2015.
- OpenAI gym provides an easy way for people to experiment with their learning agents in an array of provided toy games

OpenAI

OpenAI Gym

- Toolkit for developing and comparing reinforcement learning (RL) algorithms.
- It consists of a growing suite of environments (Atari, simulation, etc.)
  - Gym is a collection of environments/problems designed for testing and developing reinforcement learning algorithms
  - Gym is written in Python
- OpenAI Gym is compatible with algorithms written in any framework, such as **Tensorflow** and **Theano**.

Source

- **Books:**

  - R. S. Sutton and A. G. Barto, “Reinforcement Learning: An Introduction,” 2017.
  - M. W. and M. van Otterlo eds, *ALO 12 - Reinforcement Learning*. 2013.
  - V. Heidrich-Meisner, M. Lauer, C. Igel, and M. Riedmiller, “Reinforcement Learning in a Nutshell,” *Proc. 15th Eur. Symp. Artif. Neural Networks ESANN 2007*, no. April, pp. 277–288, 2007.

- **Blogs:**

  - https://dev.to/n1try/cartpole-with-q-learning---first-experiences-with-openai-gym
  - https://mpatacchiola.github.io/blog/2016/12/09/dissecting-reinforcement-learning.html
  - https://www.oreilly.com/ideas/reinforcement-learning-for-complex-goals-using-tensorflow
  - https://www.analyticsvidhya.com/blog/2017/01/introduction-to-reinforcement-learning-implementation/

- **Video Lecture:**

  - https://www.youtube.com/watch?v=w33Lplx49_A&t=288s
  - https://www.youtube.com/watch?v=jUoZg513cdE
