UNIVERSITY OF LEEDS

Class: Machine Learning

Reinforcement Learning Algorithms

Instructor: Matteo Leonetti

Learning outcomes

UNIVERSITY OF LEEDS

- Update estimates of the value function with:
  - First-visit Monte Carlo
  - Sarsa
  - Q-Learning
- Explain the difference between off and on-policy learning
- Explain the difference between Monte Carlo methods and TD learning.

Dynamic Programming

UNIVERSITY OF LEEDS

We turn the Bellmann optimality equation into an update rule:

$$q_{k+1}(s,a) = \sum_{s',r} p(s',r|s,a)\left(r + \gamma \max_{a'} q_k(s',a')\right)$$

We need a model of the dynamics and the reward of the MDP. This is *planning*.

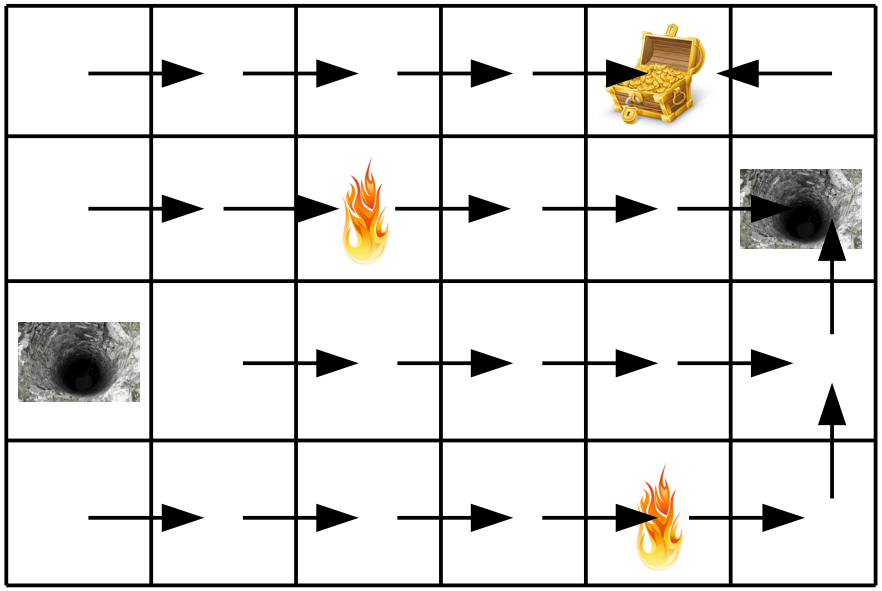
We wanted to do *learning*!

Policy iteration

UNIVERSITY OF LEEDS

Transition and reward function unknown: $MDP\ M = \langle S, A, ?, ? \rangle$

The agent starts with an arbitrary policy (or, equivalently, action-value function), For instance, this policy:

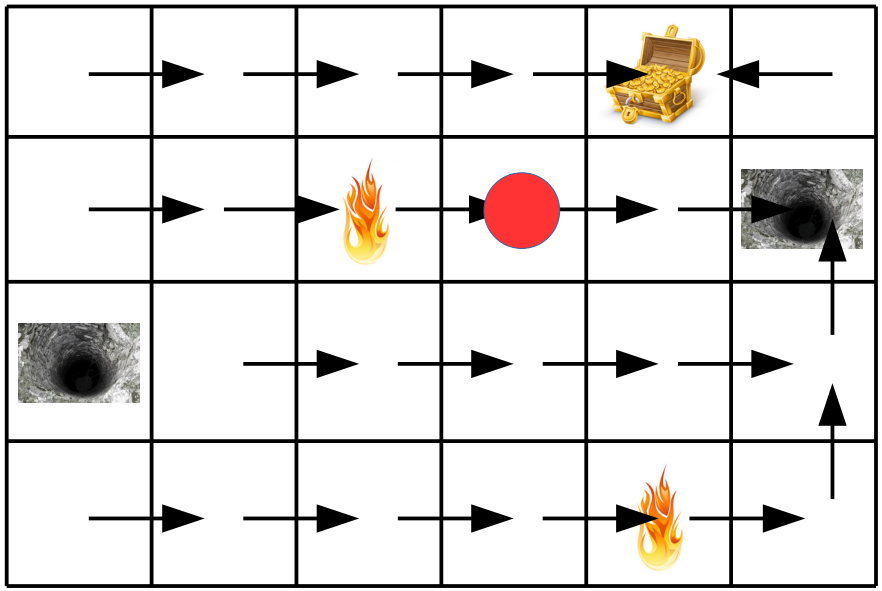


Policy iteration

UNIVERSITY OF LEEDS

Step 1: policy evaluation

Compute the action-value function of this policy



For simplicity, let's focus on one state:

$\gamma = 0.5$

$q(\langle 4,2 \rangle, \text{right}) = 0 + \gamma(-100) = -50$

$q(\langle 4,2 \rangle, \text{up}) = 0 + \gamma(50) = 25$

$q(\langle 4,2 \rangle, \text{left}) = -5 + \gamma^3(-100) = -17.5$

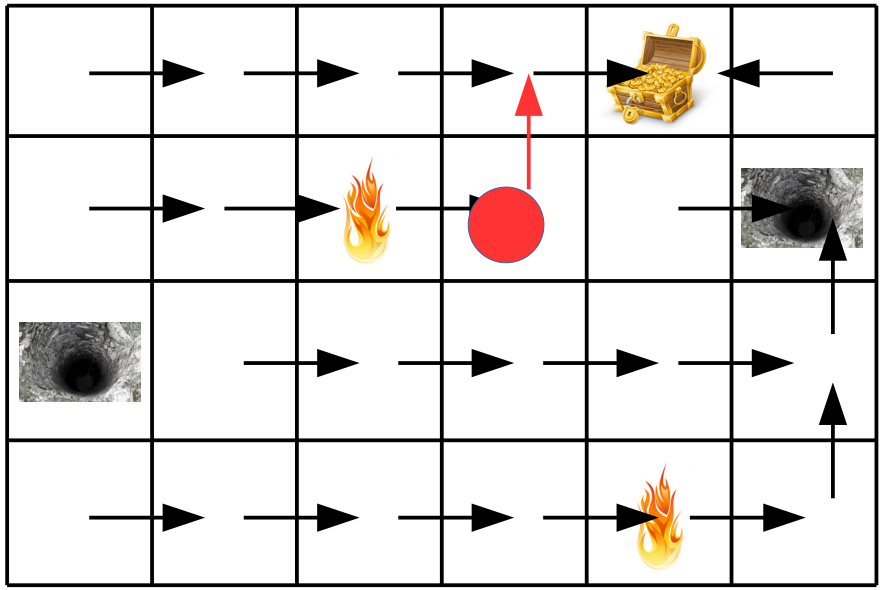
$q(\langle 4,2 \rangle, \text{down}) = 0 + \gamma^3(-100) = -12.5$

Policy iteration

UNIVERSITY OF LEEDS

Step 2: policy improvement

Having the value for the current policy, find a state in which a different action would be better than the current one.



$$q(\langle 4,2\rangle, \text{right}) = 0 + \gamma(-100) = -50$$

$$q(\langle 4,2\rangle, \text{up}) = 0 + \gamma(50) = 25$$

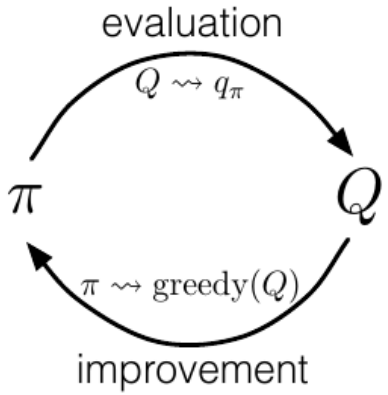
$$q(\langle 4,2\rangle, \text{left}) = -5 + \gamma^3(-100) = -17.5$$

$$q(\langle 4,2\rangle, \text{down}) = 0 + \gamma^3(-100) = -12.5$$

The current action is RIGHT, a better action is UP

Generalized Policy iteration

UNIVERSITY OF LEEDS



No need to fully evaluate a policy before making an improvement
→ *Generalized* PI

On MDPs this is a *hill-climbing* procedure, every solution is strictly better than previous ones.

How do we evaluate a policy without a model?

Estimating the value function

UNIVERSITY OF LEEDS

We are going to estimate the action-value function through successive approximations obtained by sampling from the environment.

The general learning rule will be the following:

$$q_{k+1}(s,a) = (1-\alpha)q_k(s,a) + \alpha\,\text{target}$$

$$= q_k(s,a) + \alpha(\text{target} - q_k(s,a))$$

Bellman Equation

UNIVERSITY OF LEEDS

$$q_\pi(s,a)=\sum_{s',r} p(s',r|s,a)\left(r+\gamma \sum_{a'} \pi(a'|s) q_\pi(s',a')\right)$$

Target policy: the policy I want to learn about

$$q^*(s,a)=\sum_{s',r} p(s',r|s,a)\left(r+\gamma max_{a'} q^*(s',a')\right)$$

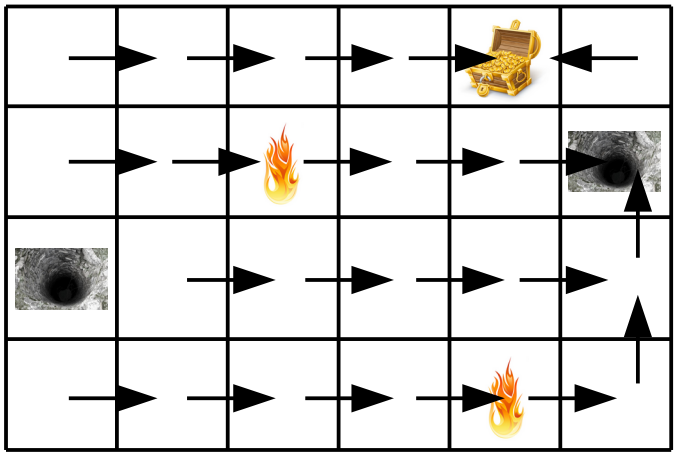
Here the target is the optimal policy

States, actions, rewards

UNIVERSITY OF LEEDS

All we have is a sequence of state, actions and rewards:

$$\langle s_{0,} a_{0,} r_{1,} s_1 \rangle, \langle s_{1,} a_{1,} r_{2,} s_2 \rangle, \langle s_{2,} a_{2,} r_{3,} s_3 \rangle, \ldots, \langle s_{n-1}, a_{n-1}, r_n, s_n \rangle$$



Example sequence:

$$\langle\langle 0,0\rangle, \mathrm{r}, 0, \langle 1,0\rangle\rangle \; \langle\langle 1,0\rangle, \mathrm{r}, 0, \langle 2,0\rangle\rangle \; \langle\langle 2,0\rangle, \mathrm{r}, 0, \langle 3,0\rangle\rangle \; \langle\langle 3,0\rangle, \mathrm{r}, -5, \langle 4,0\rangle\rangle$$

$$\langle\langle 4,0\rangle, \mathrm{r}, 0, \langle 5,0\rangle\rangle \; \langle\langle 5,0\rangle, \mathrm{u}, 0, \langle 5,1\rangle\rangle \; \langle\langle 5,1\rangle, \mathrm{u}, -100, \langle 5,2\rangle\rangle$$

First-visit Monte Carlo

UNIVERSITY OF LEEDS

A Monte-Carlo update uses a sample of the return as the target:

The return: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$

$\langle\langle 0,0\rangle, \mathrm{r}, 0, \langle 1,0\rangle\rangle \; \langle\langle 1,0\rangle, \mathrm{r}, 0, \langle 2,0\rangle\rangle \; \langle\langle 2,0\rangle, \mathrm{r}, 0, \langle 3,0\rangle\rangle \; \langle\langle 3,0\rangle, \mathrm{r}, -5, \langle 4,0\rangle\rangle$

$\langle\langle 4,0\rangle, \mathrm{r}, 0, \langle 5,0\rangle\rangle \; \langle\langle 5,0\rangle, \mathrm{u}, 0, \langle 5,1\rangle\rangle \; \langle\langle 5,1\rangle, \mathrm{u}, -100, \langle 5,2\rangle\rangle$

$$q^{k+1}(\langle 0,0\rangle, \mathrm{r}) = q^k(\langle 0,0\rangle, \mathrm{r}) + \alpha\left(0 + \gamma\cdot 0 + \gamma^2\cdot 0 + \gamma^3\cdot -5 + \dots - q^k(\langle 0,0\rangle, \mathrm{r})\right)$$

target

...

$$q^{k+1}(\langle 5,0\rangle, \mathrm{u}) = q^k(\langle 5,0\rangle, \mathrm{u}) + \alpha\left(0 + \gamma\cdot(-100) - q^k(\langle 5,0\rangle, \mathrm{u})\right)$$

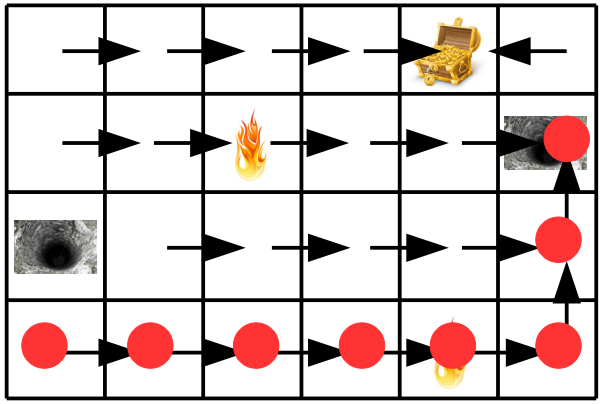
$$q^{k+1}(\langle 5,1\rangle, \mathrm{u}) = q^k(\langle 5,1\rangle, \mathrm{u}) + \alpha\left(-100 - q^k(\langle 5,1\rangle, \mathrm{u})\right)$$

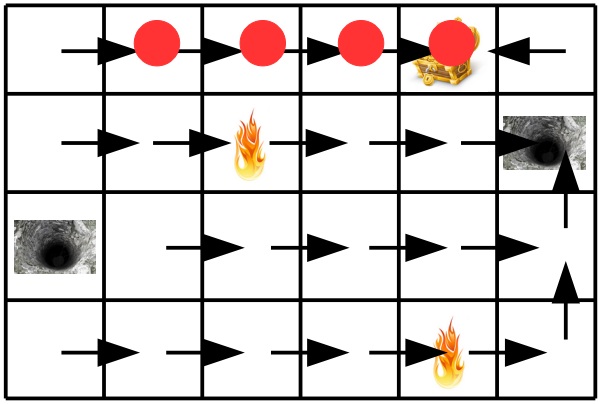
Exploration

UNIVERSITY OF LEEDS

The agent needs to cover every state action pair.

One option (when possible) is to do random restarts:



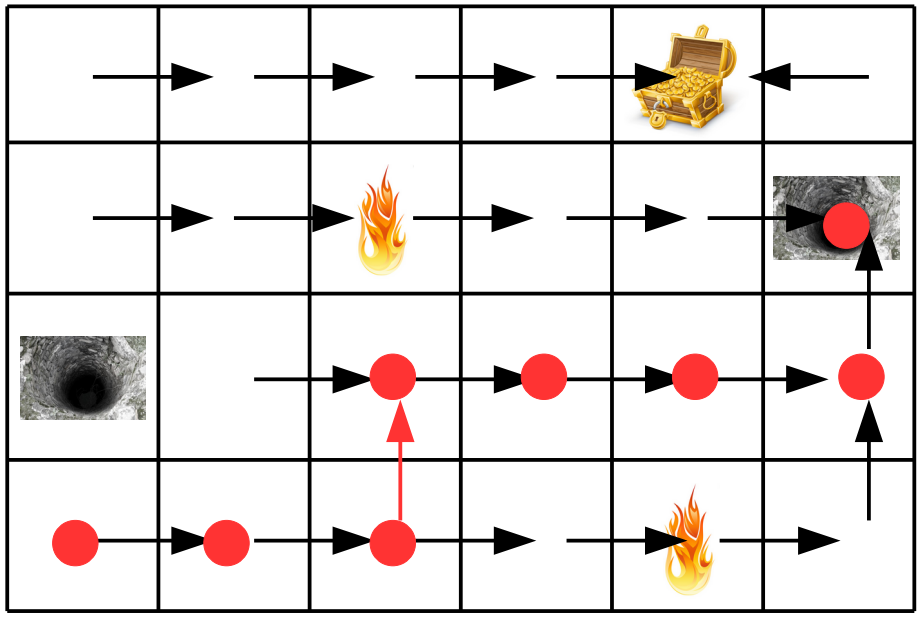


How can the agent estimate the value of the actions that the policy does not take?

The other option is to *explore*: every once in a while take an action different from the policy under evaluation.

Exploration vs. exploitation

UNIVERSITY OF LEEDS



Exploratory step (in red)

Exploration-exploitation tradeoff

ε-greedy

UNIVERSITY OF LEEDS

For ease of implementation, we will use a very simple rule for exploration called ε-greedy.

At every time step:

Follow the current optimal policy with probability $(1-\epsilon)$

Take a random action with probability: ϵ

On Monte Carlo updates

UNIVERSITY OF LEEDS

Monte Carlo updates can only be done after a whole sequence has been collected.

Problem #1: valid only for episodic tasks

Problem #2: it cannot learn **during** an episode

TD methods

UNIVERSITY OF LEEDS

We address those two problems with a new class of algorithms called *temporal difference* methods.

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = R_{t+1} + \gamma G_{t+1}$$

$$q_\pi(s,a) = E_\pi[G_t | S_t = s, A_t = a] \leftarrow \text{Monte-Carlo target}$$

$$= E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a]$$

$$= E_\pi[R_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k R_{t+k+2} | S_t = s, A_t = a]$$

$$= E_\pi[R_{t+1} + \gamma q(S_{t+1}, A_{t+1}) | S_t = s, A_t = a] \leftarrow \text{TD target}$$

Targets

UNIVERSITY OF LEEDS

TD target

$$E[R_t + \gamma q(S_{t+1}, A_{t+1}) | S_t, A_t] = \sum_{s'} p(s'|S_t, A_t)(R_{t+1} + \gamma \sum_{a'} \pi(a'|s') q(s', a'))$$

Two distributions to sample from:

Sarsa On-policy

$$q_{k+1}(s,a) = q_k(s,a) + \alpha(r_{t+1} + \gamma q_k(S_{t+1}, A_{t+1}) - q_k(s,a))$$

Expected Sarsa On or off-policy

$$q_{k+1}(s,a) = q_k(s,a) + \alpha(r_{t+1} + \gamma \sum_{a'} \pi(a'|s') q(s', a') - q_k(s,a))$$

Q-learning Off-policy

$$q_{k+1}(s,a) = q_k(s,a) + \alpha(r_{t+1} + \gamma max_{a'} q(s', a') - q_k(s,a))$$

Sarsa

UNIVERSITY OF LEEDS

$$\text{target} = R_{t+1} + \gamma\, q_\pi(S_{t+1}, A_{t+1})$$

$$q_\pi^{k+1}(s,a) = q_\pi^k(s,a) + \alpha\left(r_{t+1} + \gamma\, q_\pi^k(s', \pi(s')) - q_\pi^k(s,a)\right)$$

Given only one step from the sequence from the example in slide 13

$$\langle\langle 2,0\rangle, \mathrm{r}, 0, \langle 3,0\rangle\rangle \quad \langle\langle 3,0\rangle, \mathrm{r}, -5, \langle 4,0\rangle\rangle$$

$$q_\pi^{k+1}(\langle 2,0\rangle, \mathrm{r}) = q_\pi^k(\langle 2,0\rangle, \mathrm{r}) + \alpha\left(0 + \gamma\, q_\pi^k(\langle 3,0\rangle, \mathrm{r}) - q_\pi^k(\langle 2,0\rangle, \mathrm{r})\right)$$

Bootstrapping

UNIVERSITY OF LEEDS

Monte Carlo

$$q_\pi^{k+1}(s,a) = q_\pi^k(s,a) + \alpha\left(g_t - q_\pi^k(s,a)\right)$$

g_t is a sample of the return G_t

Sarsa

$$q_\pi^{k+1}(s,a) = q_\pi^k(s,a) + \alpha\left(r_{t+1} + \gamma q_\pi^k(s',\pi(s')) - q_\pi^k(s,a)\right)$$

another prediction

TD methods *bootstrap:* the new prediction of the expected return is based not only on data (as in Monte Carlo) but also on another prediction.

Q-learning

UNIVERSITY OF LEEDS

Very similar to Sarsa, indeed it is another TD method

$$\text{target} = R_{t+1} + \gamma\, q_\pi(S_{t+1}, A_{t+1})$$

$$q_\pi^{k+1}(s,a) = q_\pi^k(s,a) + \alpha\left(r_{t+1} + \gamma\, max_{a'}\, q_\pi^k(s',a') - q_\pi^k(s,a)\right)$$

Next action not from the policy: Q-learning is *off-policy*

On vs Off-policy

UNIVERSITY OF LEEDS

On-policy methods, such as Monte-Carlo and Sarsa, learn the value of the policy that is generating the samples.

In order to converge to the optimal policy, such a policy must be iteratively improved (cf. the policy improvement step).

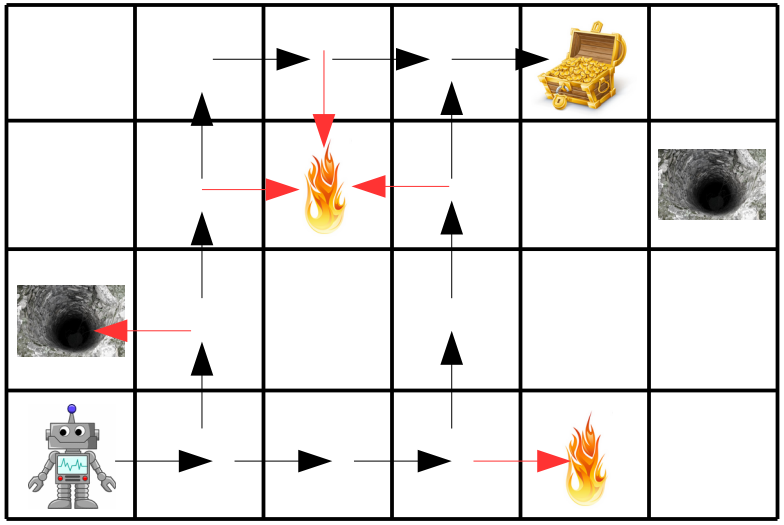
Off-policy methods, such as Q-learning, learn the value function of the optimal policy **directly** (they are the learning version of value iteration in dynamic programming).

Understanding On vs Off-policy

UNIVERSITY OF LEEDS

Agent acting according to an ε-greedy policy.

Do Q-learning and Sarsa learn the same policy?

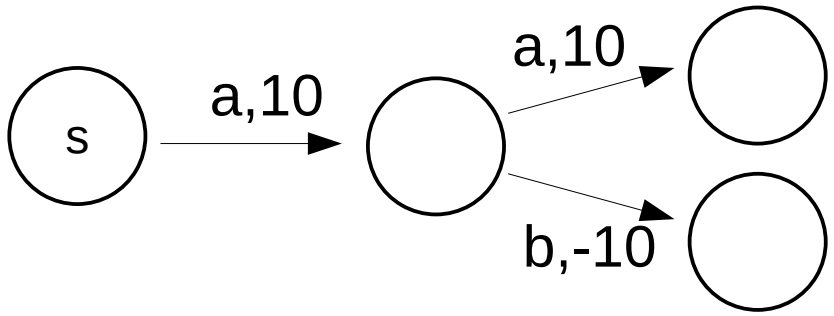


A simple numerical example

UNIVERSITY OF LEEDS

Agent acting according to an ε-greedy policy.

$\epsilon = 0.1 \qquad \gamma = 0.5$



Action value learned by Sarsa:

$$q(s,a) = 10 + \gamma(0.9 \cdot 10 + 0.1 \cdot (-10)) = 14$$

Action value learned by Q-learning:

$$q(s,a) = 10 + \gamma(10) = 15$$

History

UNIVERSITY OF LEEDS

- 1957 - Richard Bellman (Optimal Control thread)

- 1911 - Edward Thorndike (Psychology thread)

History

UNIVERSITY OF LEEDS

- (since) 1970s – Andy Barto (UMass Amherst) and Rich Sutton (U of Alberta)

Authors of the main RL textbook: Reinforcement Learning: an Introduction.

For most of us “the book” (http://incompleteideas.net/book/the-book-2nd.html).

- 1989 – Chris Watkins (U of London)

Q-learning (http://www.cs.rhul.ac.uk/home/chrisw/)

History

UNIVERSITY OF LEEDS

- 1992 – Gerry Tesauro

(… a lot of other stuff…)

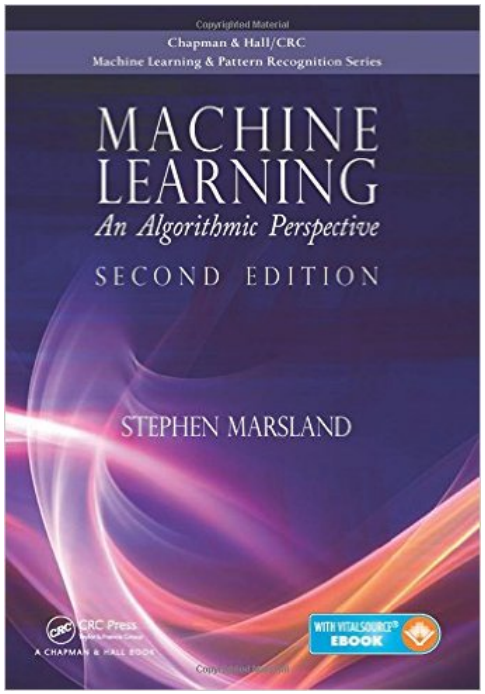
- 2013 – A lot of people…

DeepMind

I'll mention David Silver as you have probably heard of AlphaGo (2015)

Conclusion

UNIVERSITY OF LEEDS



Chapter 11