Introduction

Herke van Hoof

About reinforcement learning

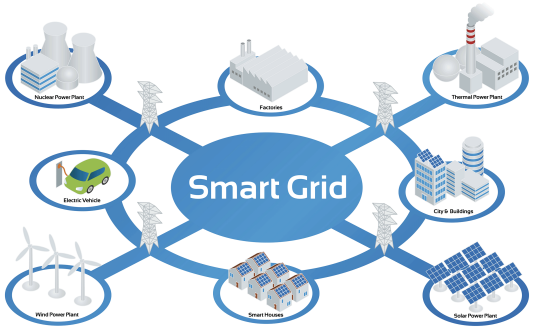
How to **sequentially interact**
with an environment to
maximise a long-term objective?

About reinforcement learning

How to **sequentially interact** with an environment to **maximise a long-term objective**?

Why is this important?

About reinforcement learning



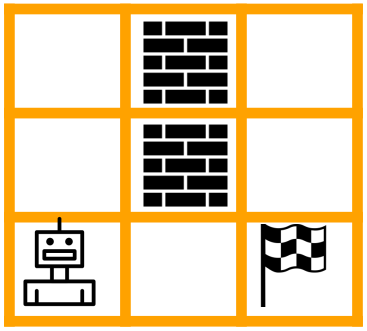
www.switchmybusiness.com

robots.nu

www.energyfuse.org

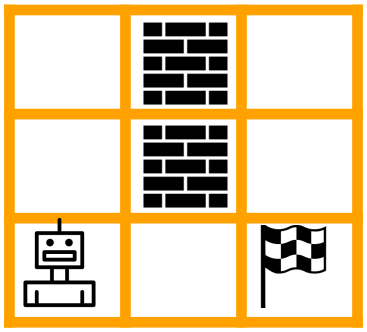
robotshop.com

About reinforcement learning



state
‘chosen’ by environment

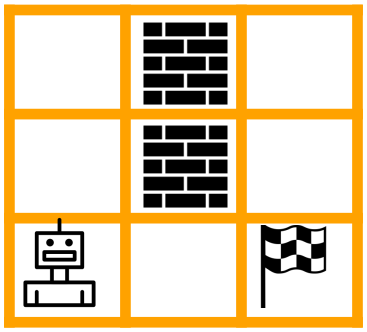
About reinforcement learning

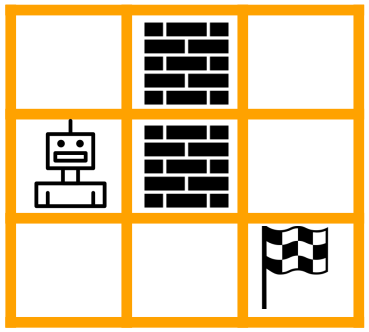


state
'chosen' by environment

action
chosen by agent

About reinforcement learning





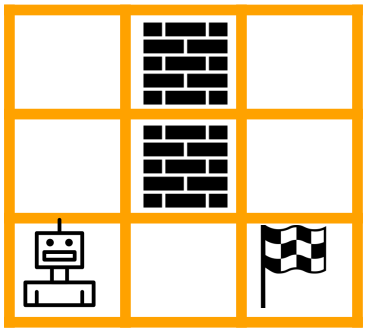
state
'chosen' by environment

action
chosen by agent

reward
'chosen' by environment

next state

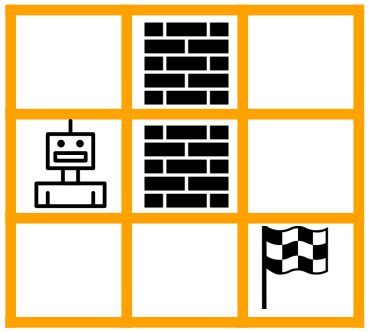
About reinforcement learning



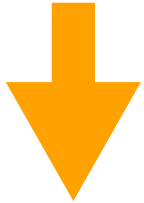
state
'chosen' by environment

action
chosen by agent

reward
'chosen' by environment



next state



next
action?

About reinforcement learning

Machine learning

Supervised learning

Unsupervised learning

‘2’

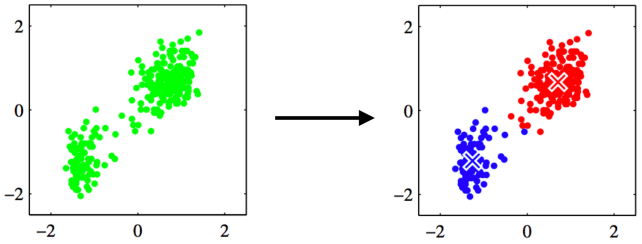


Figure: C.M. Bishop, Pattern recognition & Machine learning

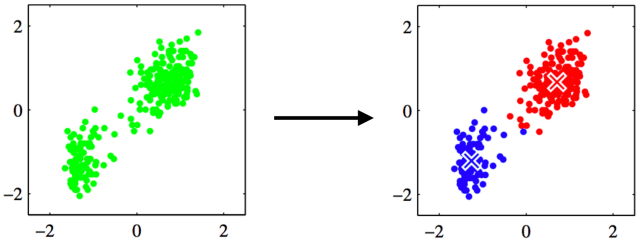
About reinforcement learning

Machine learning

Supervised learning

Unsupervised learning

‘2’

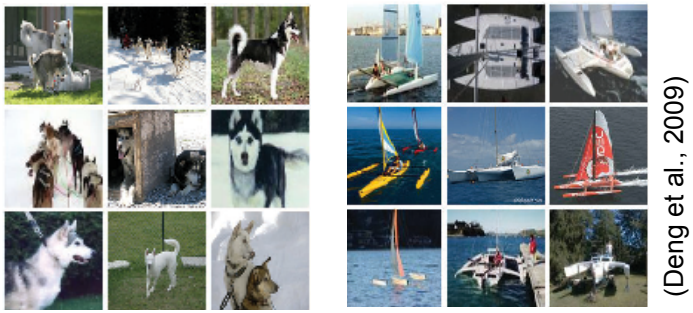


Reinforcement learning
?

Figure: C.M. Bishop, Pattern recognition & Machine learning

Learning to do vs learning to predict

Supervised learning



(Deng et al., 2009)

Learn to predict

Dataset with human response

Data doesn't depend on what is learned

Data are iid

Learning to do vs learning to predict

Supervised learning

(Deng et al., 2009)

Learn to predict

Dataset with human response

Data doesn't depend on what is learned

Data are iid

Reinforcement learning

robotshop.com

Learn to do

Improve over human response

Data depends on current strategy

Data are sequential

About the course

Before we dive into any details, let's look at what the course is going to be like!

Course format

Lectures:

- Preferrably in-person. Better interaction & adjusting to group
- Recordings available via canvas

Tutorial sessions

- Fully on campus

Tutorials & Exercises

Practice is essential for learning!

- *ungraded exercises* to practice the material
- Some questions from last years' exams, shows level & type of questions
- Roughly, 1st hour tutorial: work on ungraded exercises together

- Every week, written homework & programming assignment(s)
- Homework questions tend to be a bit easier than exam questions
- Roughly, 2nd hour tutorial: work on assignments, can ask questions

How to do well in the course

Attend the lecture and review before tutorial session

Actively participate in tutorial classes. You can benefit from interaction with TAs and other students

Don’t skip the ungraded exercises!

Reading material offers additional insights and different perspectives, suggested to read alongside lectures.

About the course

Recordings, slides, and assignments will be posted on Canvas!

Important announcements will be send from Canvas as well - make sure they reach you!

Also, you'll find the syllabus with:

- Reading suggestions per week
- Deadlines
- Organisational information

About the course

Week	Topic	
1	Recap RL basics	
2	Tabular methods	HW 1 due
3	Learning with approximation	HW 2 due
4	With approximation & direct policy learning	HW 3 due
5	Advanced direct policy learning	HW 4 due
6	Planning, guest lecture	HW 5 due
7	Partial observability, recap	HW 6 due
8	**Exam!**	

About the course

Week	Topic	
1	Recap RL basics	RL basics
2	Tabular methods	
3	Learning with approximation	
4	With approximation & direct policy learning	
5	Advanced direct policy learning	Advanced topics
6	Planning, guest lecture	
7	Partial observability, recap	
8	**Exam!**	

About the course

Week	Topic	
1	Recap RL basics	RL basics
2	Tabular methods	
3	Learning with approximation	
4	With approximation & direct policy learning	
5	Advanced direct policy learning	Advanced topics
6	Planning	
7	Partial ob	
8	**Exam!**	

Some activities in different room!

Prior knowledge

Working knowledge of machine learning: least-squares, neural networks, gradient descents

The basics: calculus, linear algebra, probability, statistics, programming (Python)

Workload

Total: 168 hours (6 credits x 28 hours): 21 hrs/week

Lectures: 28 hours

Tutorial session: 28 hours

Reading (lecture and exam prep): 48 hrs (avg. *6 hrs/week*)

Assignments: 71 hours (avg *9 hrs/week* + class time)

Assessment

5 coding assignments. 2% each

- Suggested in pairs (individual possible)
- Hand-in online as instructed. Use feedback from codegra.de!
- No extensions will be given

5 exercise sets. 4% each

- Suggested in pairs (individual possible)
- Hand-in online on Canvas.
- No extensions will be given

Reproducibility report. 5%

- Suggested in pairs (individual possible)
- To be handed in in week 7

Exam. 65%

What can you expect of us?

We are happy to receive any feedback

- Just get in touch with me via e-mail, Canvas, Piazza, or talk to me in the break!

I appreciate getting feedback via UvA-Q

- It helps me to improve the course
- Please fill out the questionnaire you'll get during the exam

What can you expect of the course?

RL is a complex: Even basics require quite a bit of nuance

Basics essential for understanding new papers

We'll touch on state of the art, but focus will be on solid basis

I'll illustrate underlying principles through algorithms

Thus, **we'll cover quite many algorithms**. Not all are used in practice, but they help understand fundamentals.

No need to know all algorithms by heart.

Important part: understand the principles that distinguish them.

Study materials

Main resource:

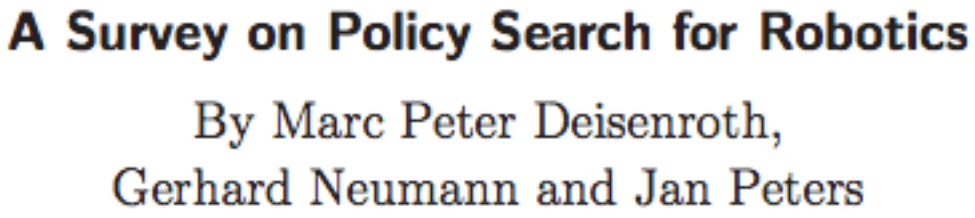
Reinforcement learning:
An Introduction (RL:AI)
R. S. Sutton & A.G. Barto

We use 2nd edition, it is also available for free online (link on canvas)

(Most topics are also described in the old book)

Study materials

Our discussion of policy based methods will be based on the survey by Deisenroth, Neumann & Peters (link on Canvas)



We will also look at additional papers (links will be distributed on Canvas)

The additional papers help understand the lecture content, but details not covered in the lecture will not be on the exam.

Study materials

Slides will be available as soon as possible after the lecture

Slides not intended as self-contained study material. For self-study the recommended reading from book is better suited

Exam will cover all material (lectures, reading material, assignments)

Questions about course organization?

About us

Instructor: Herke van Hoof

- Assistant professor in the Amsterdam machine learning lab (AMLab)
- Background: machine learning for robots
- Research focus:

  Reinforcement learning with structured solution spaces

  Reinforcement learning for combinatorial problem solving

Seethu Christopher

- Lecturer at GSI, UvA; Researcher at HIT, Delft
- Research Focus: Social Robotics for Healthcare, Rehabilitation Robotics, Healthcare Robotics

About us

Dimitris Michailidis
PhD student at Civic AI lab
Research Focus: ML to design inclusive cities

Mayesha Tasnim
PhD student at Civic AI lab
Research focus: Reinforcement mechanism design

Niklas Höpner
PhD student at AMLab
Research focus: Deep learning for Human-AI interaction

Rob Romijnders
PhD student at AMLab
Research Focus: Bayesian Learning

Stefanos Achlatis
PhD student at VIS Lab
Research focus: Offline RL and RL in the Medical Domain

Groups

Group assignments should avoid conflicts as much as possible

In principle, we'll only **switch** people between groups

- Can post a request on Piazza
- Send Herke an e-mail with the names of the two people (with e-mail addresses and student IDs) that want to switch.

Contacting us

Questions about lecture content:

- **Piazza**
- **During lecture or break**

Questions regarding exercises & practicals, (inc. grading) should be send to the **TA of your group (or Piazza)**

Feedback on lecture content

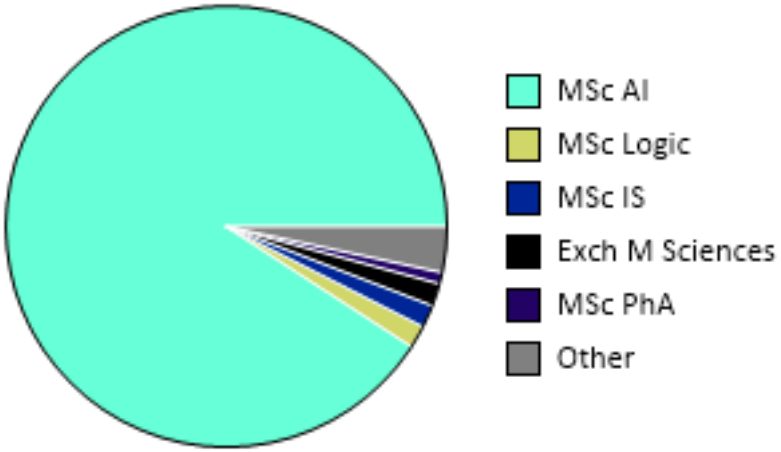
- Canvas message or email h.c.vanhoof@uva.nl

Confidential matters or when an issue cannot be resolved by TA

- Canvas message or email h.c.vanhoof@uva.nl

We try to answer all questions soon, but it might take us a day or two to reply

About you



Which BSc?
Prior RL knowledge?

All ears

https://www.allearsamsterdam.com/

If you encounter any problems during programme related event or a course and you want to file a complaint or submit separate feedback, please do not hesitate to contact the programme committee at: ocai-science@uva.nl

For more information, please see this page.

Models of decision processes

k-armed bandits

- a simple model for decision making

Markov decision processes

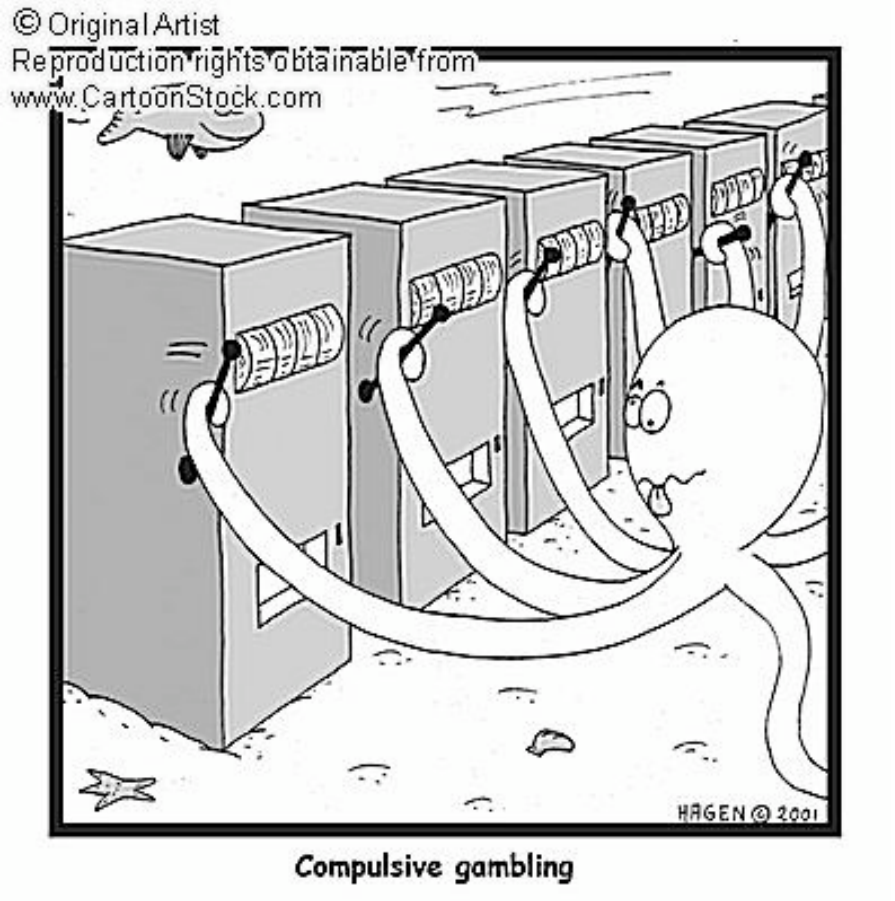
- a model for sequential decision processes

k-armed bandits

There are *k* slot machines to choose from

Each machine has an unknown distribution of payoffs

Goal: maximize cumulative payoff over some period



Thanks to Shimon Whiteson

k-armed bandits

Formally:

- There are *k* actions available at any time step
- After action a_t, receive reward r_t from unknown distribution $p(r_t|a_t)$
- Two types of objectives:

  Finite horizon,
  maximise total reward over T actions

$$\sum_{t=1}^{T} r_t$$

  Infinite horizon,
  maximise total discounted rewards
  with discount factor $\gamma \in [0,1)$

$$\sum_{t=1}^{\infty} \gamma^t r_t$$

k-armed bandits

Formally:

- There are *k* actions available at any time step
- After action a_t, receive reward r_t from unknown distribution $p(r_t|a_t)$
- Two types of objectives:

  Finite horizon,
  maximise total reward over T actions

$$\sum_{t=1}^{T} r_t$$

  Infinite horizon,
  maximise total discounted rewards
  with discount factor $\gamma \in [0,1)$

$$\sum_{t=1}^{\infty} \gamma^t r_t$$

What happens if $\gamma=1$ with an infinite horizon?

How to estimate value of each arm?

The value of an action is the expected reward

$$Q_t(a) = \mathbb{E}[r_t | a_t]$$

Estimating an expectation is easy; if action a has been chosen k_a times, yielding rewards $r_1, \ldots r_{ka}$:

$$\hat{Q}_t(a) = \frac{\sum_{i=1}^{k_a} r_i}{k_a}$$

But this requires storing all rewards, ever…

How to estimate value of each arm?

Incremental solution:

The average age of 9 people is 20
We add a 10th person who is 21 years old

What is the average of the group now?

How to estimate value of each arm?

Incremental solution:

The average age of 9 people is 20
We add a 10th person who is 21 years old

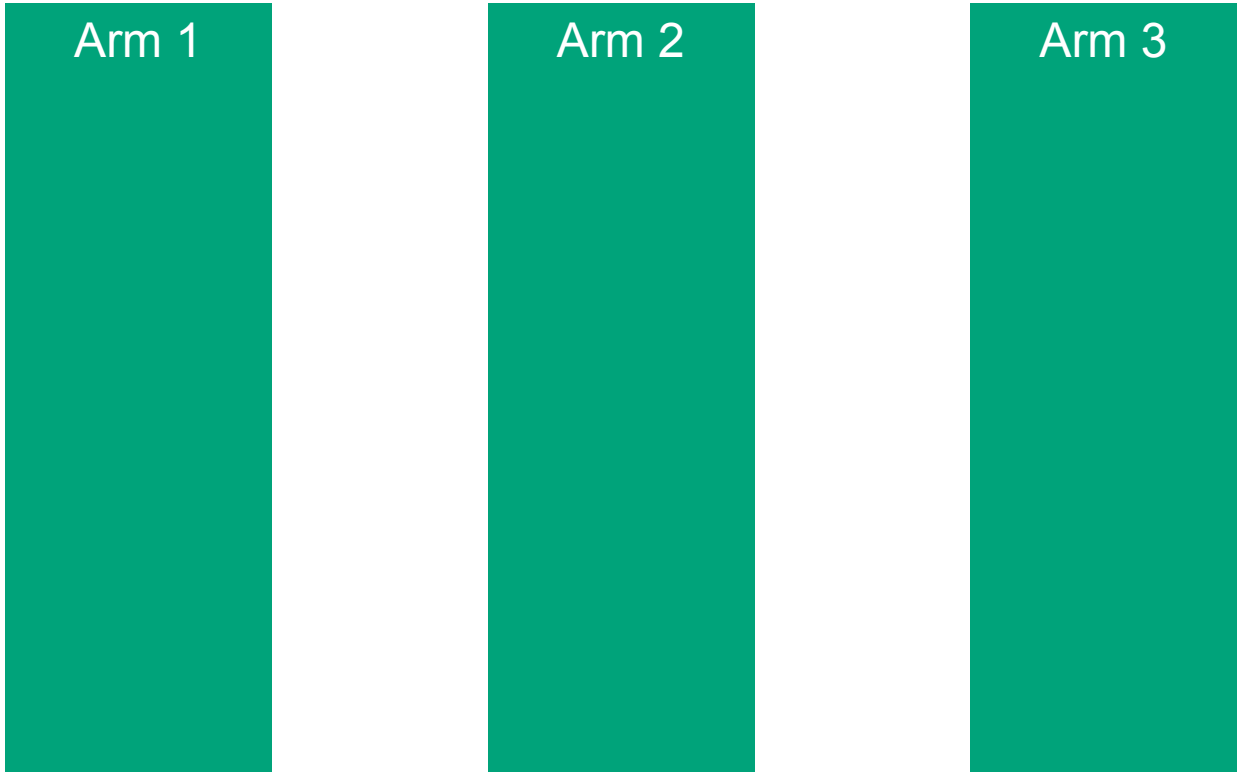
What is the average of the group now?

Similarly for the Q-function

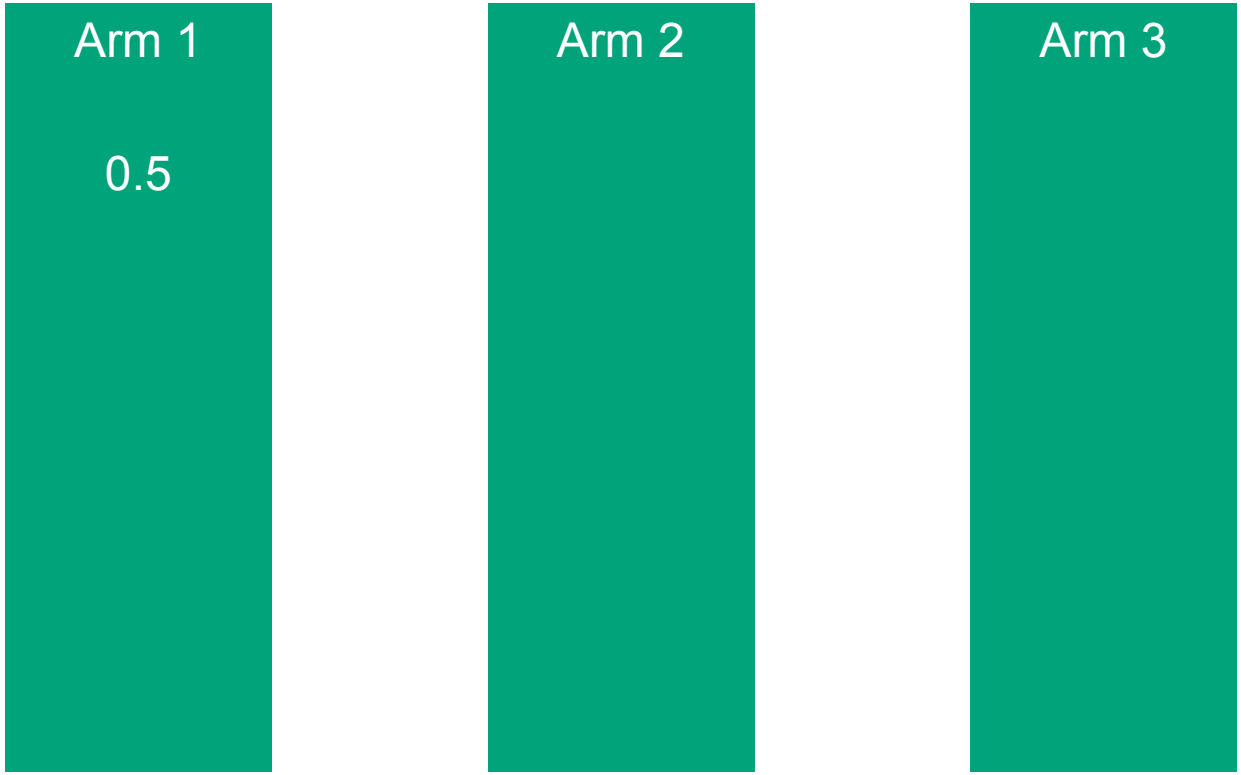
$$\hat{Q}_{t+1}(a) = \hat{Q}_t(a) + \frac{1}{k_a + 1}[r_t - \hat{Q}_t(a)]$$

#times before

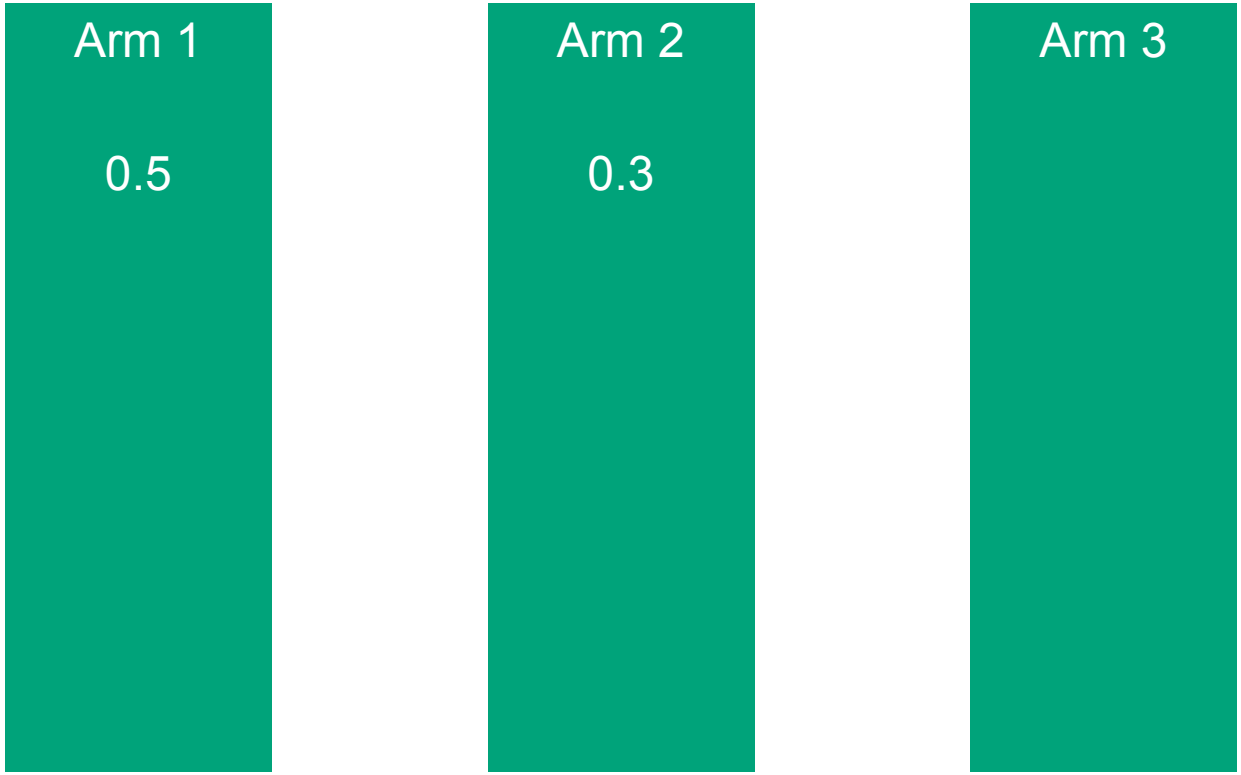
Example



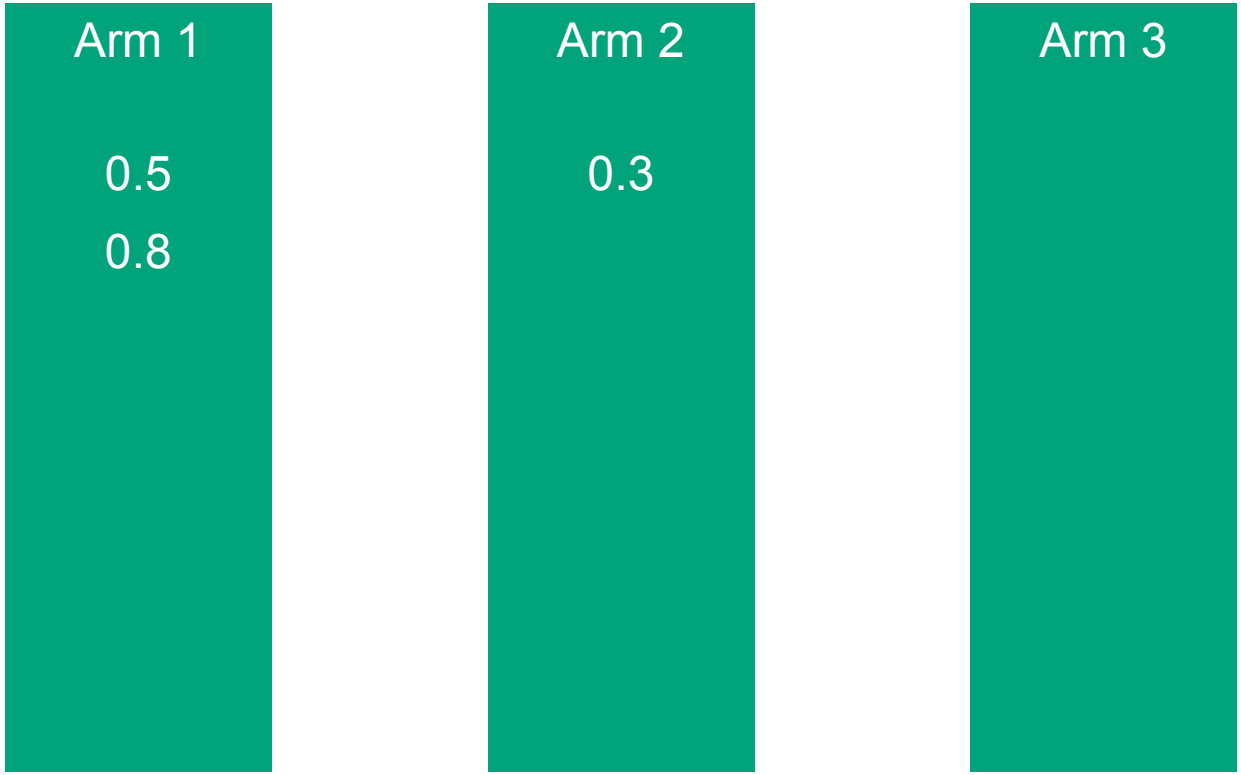
Example



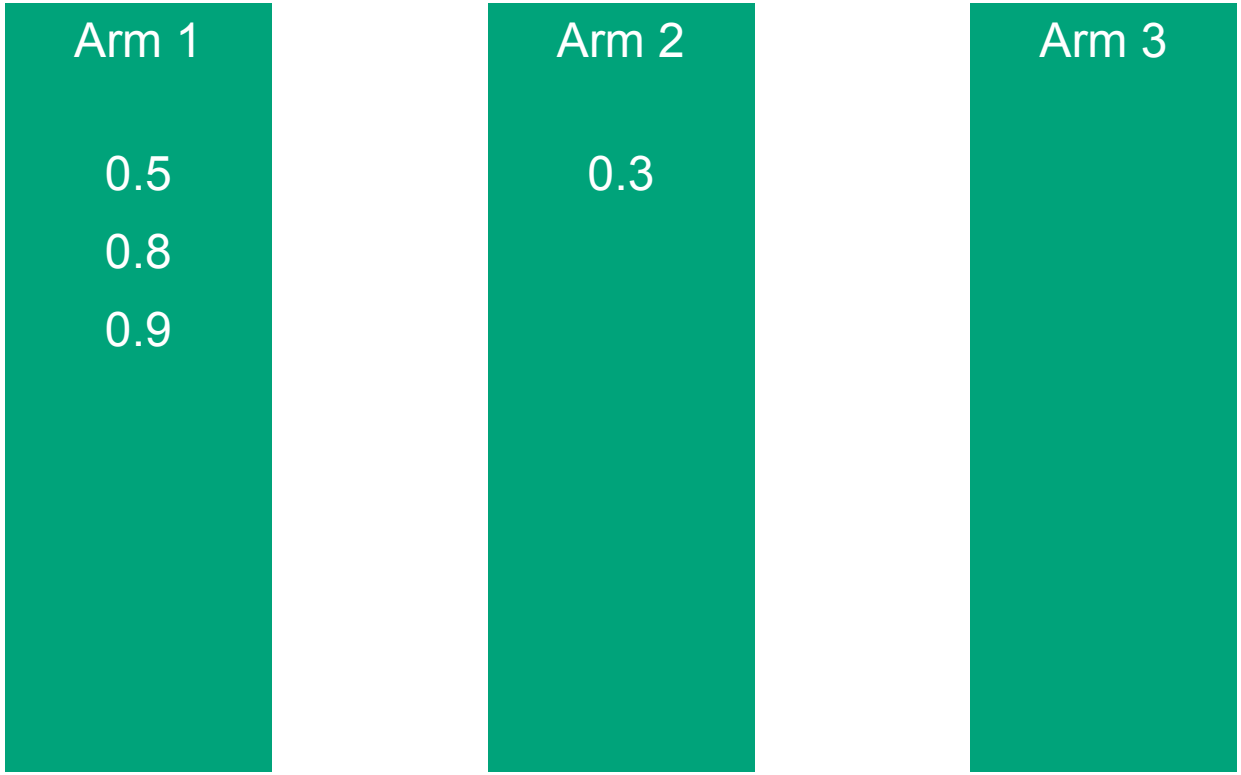
Example



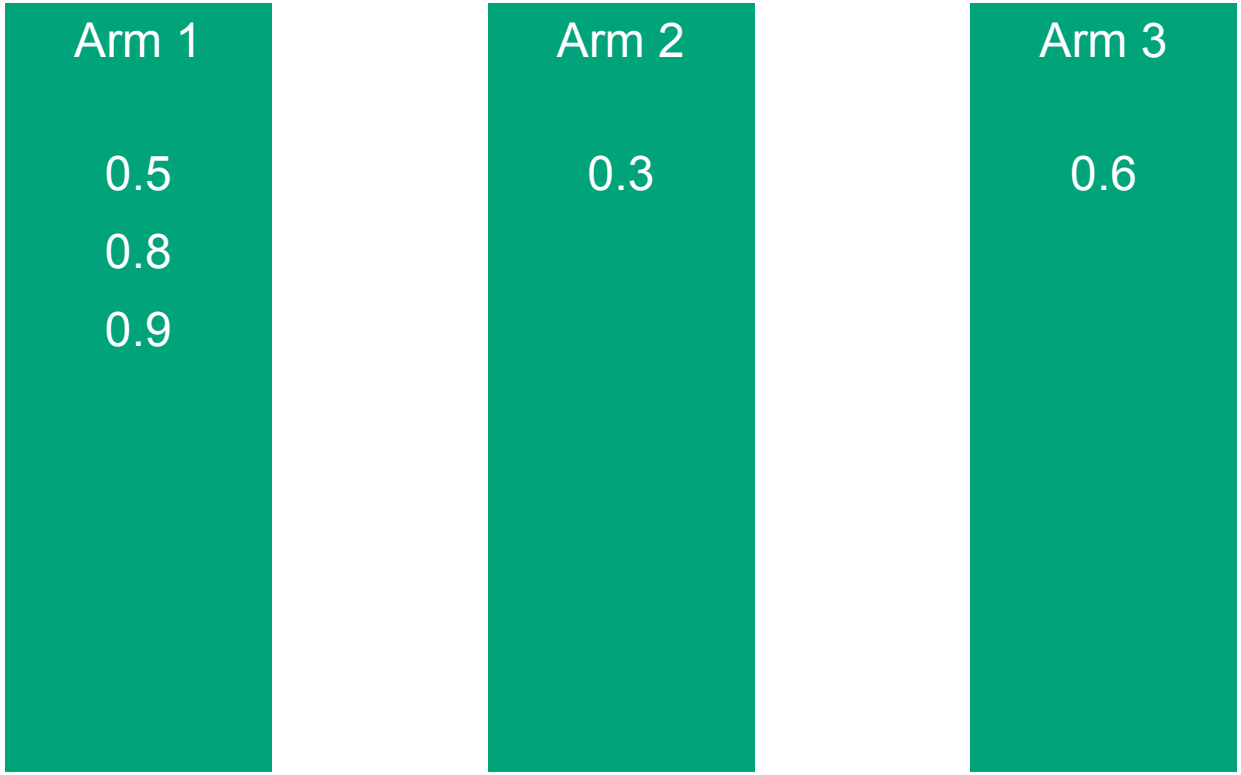
Example



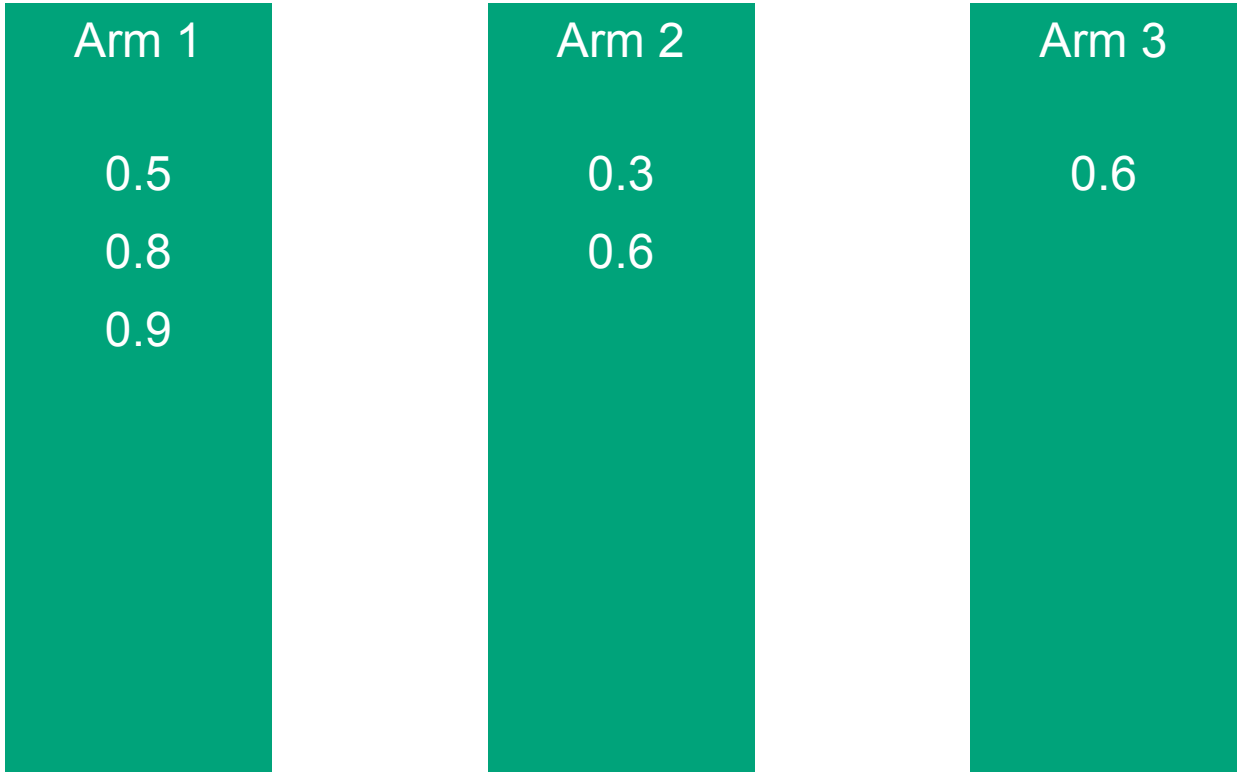
Example



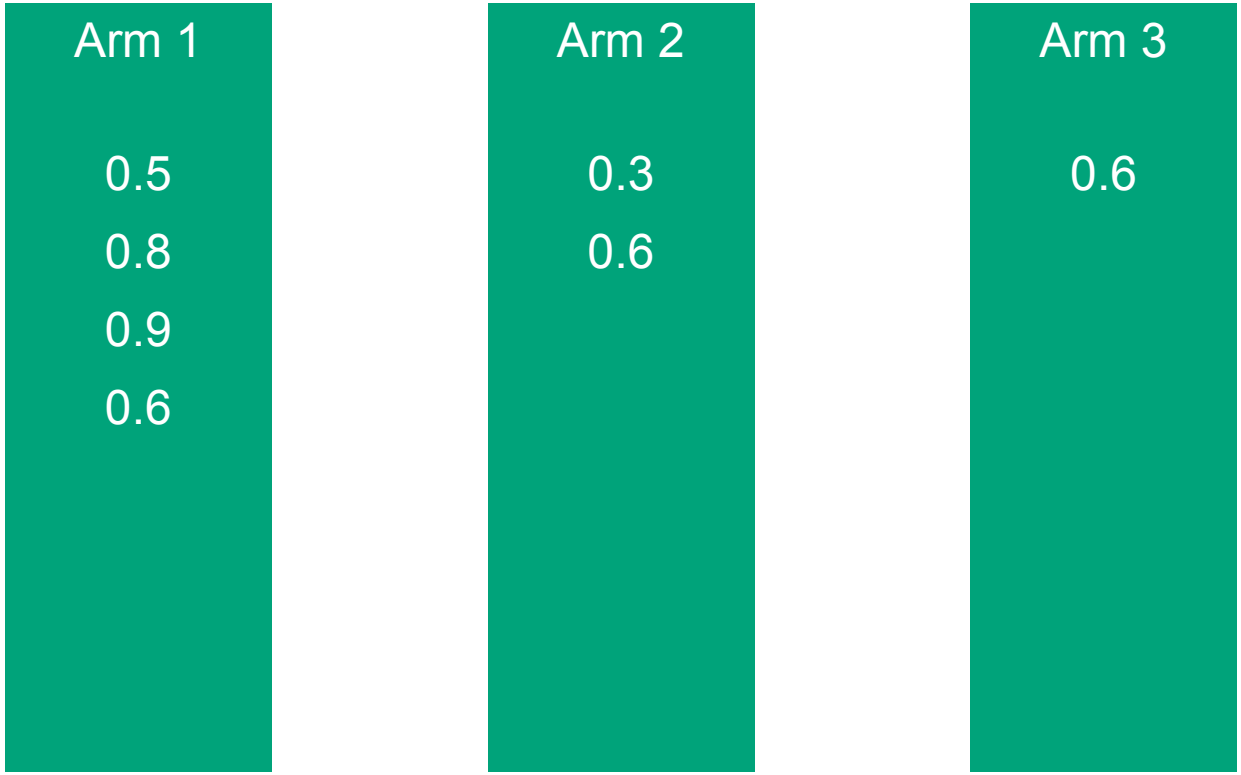
Example



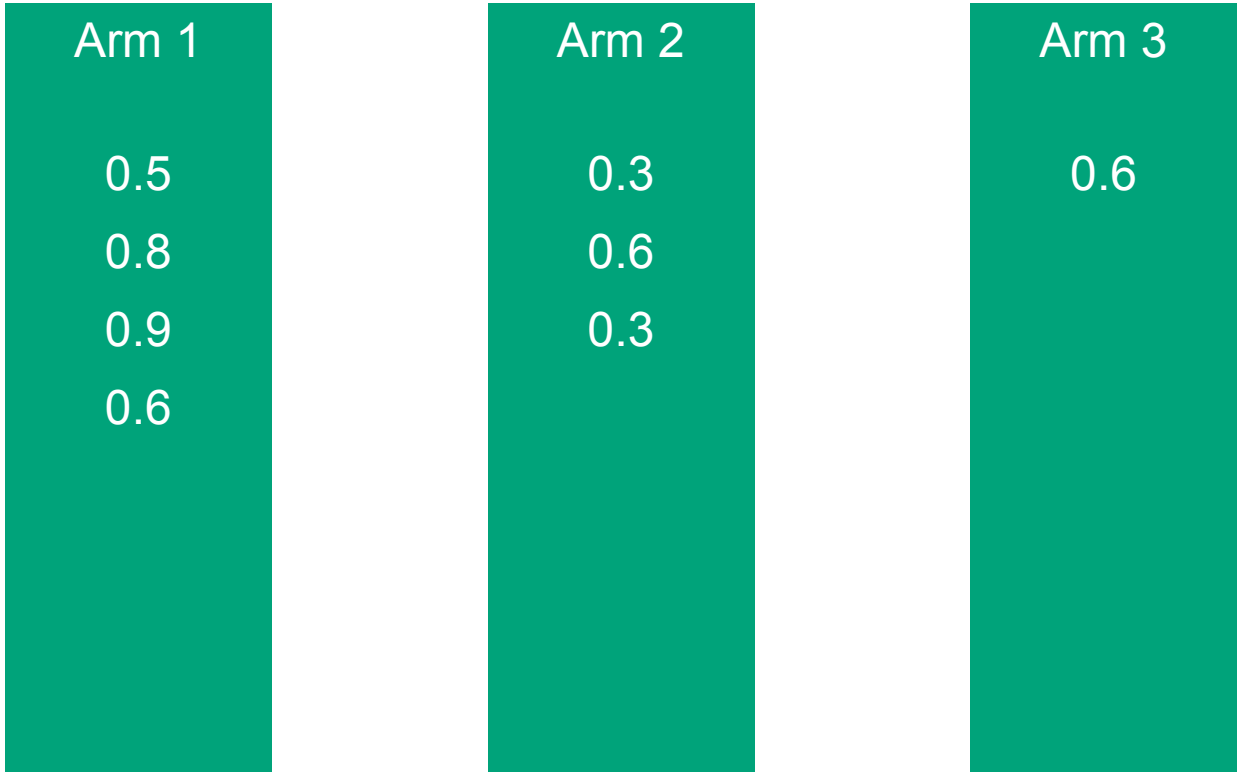
Example



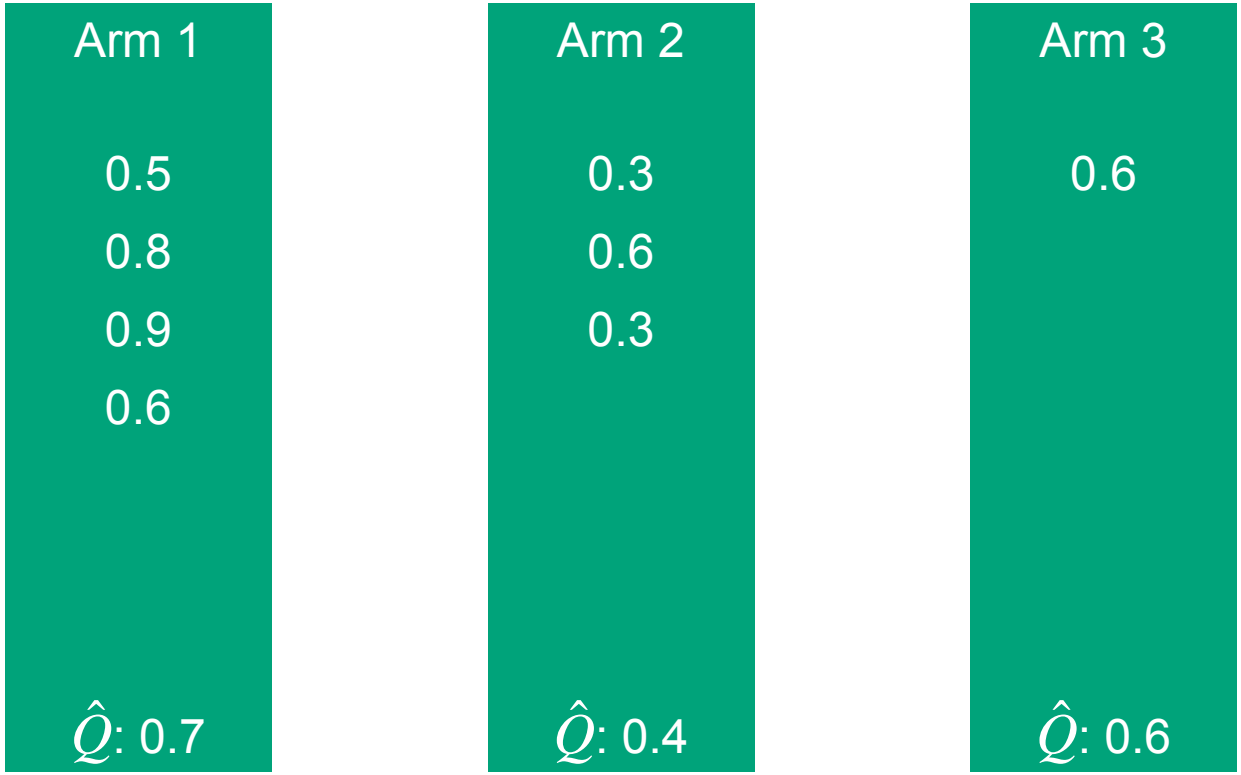
Example



Example



Example



Exploration and exploitation

If we only care about getting the best reward now, we can pick the arm with highest average

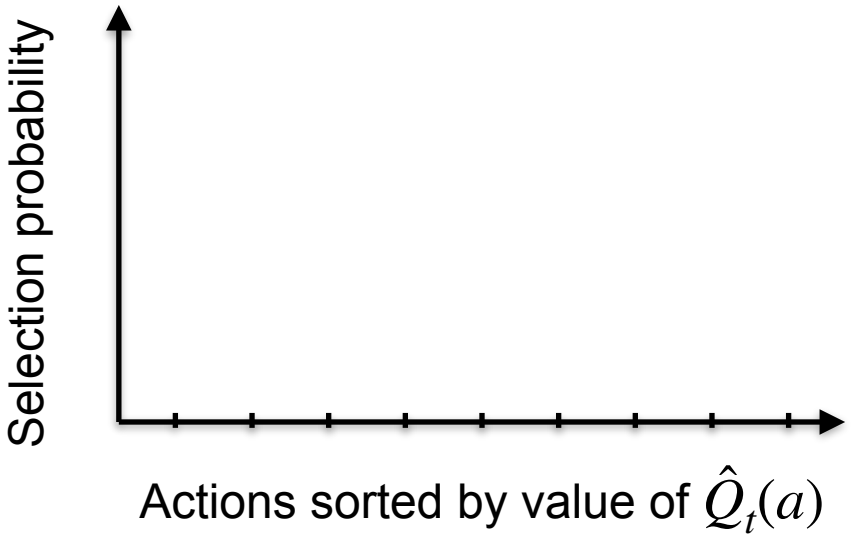
However, we might overlooked an even better action that we underestimate, and we will never find it by being greedy

Much time left / less certain: do more exploration

Less time left / more certain: do more exploitation

Exploration strategies

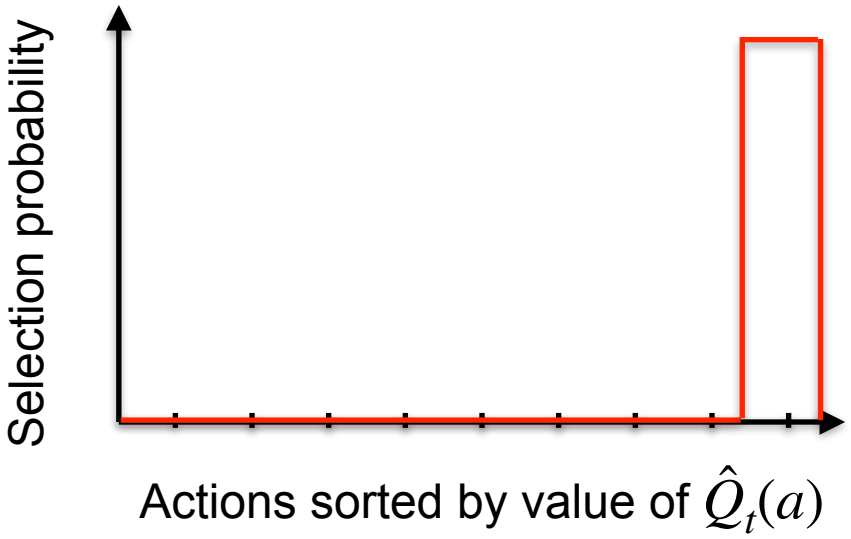
No exploration: greedy strategy



Figures: Sutton&Barto, RL:AI

Exploration strategies

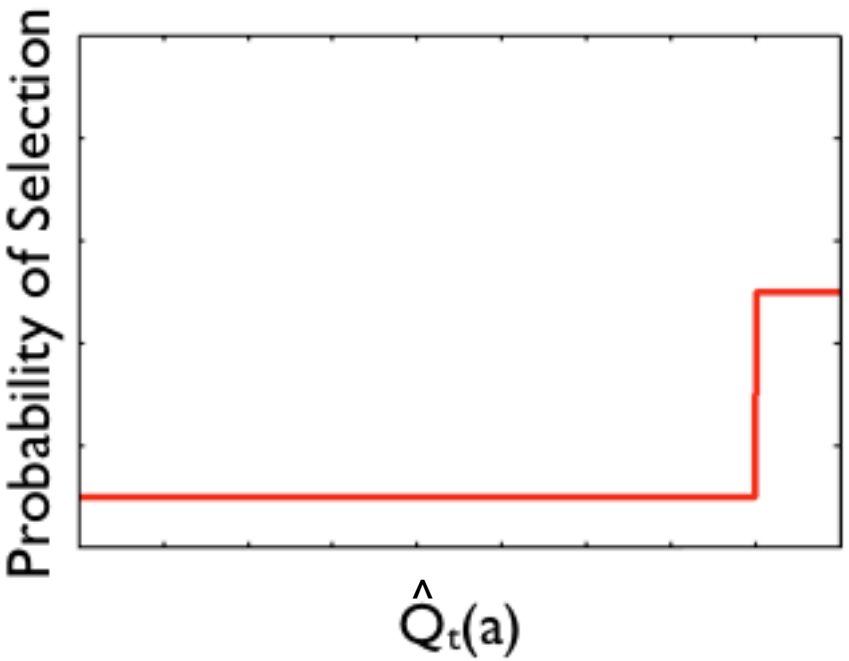
No exploration: greedy strategy

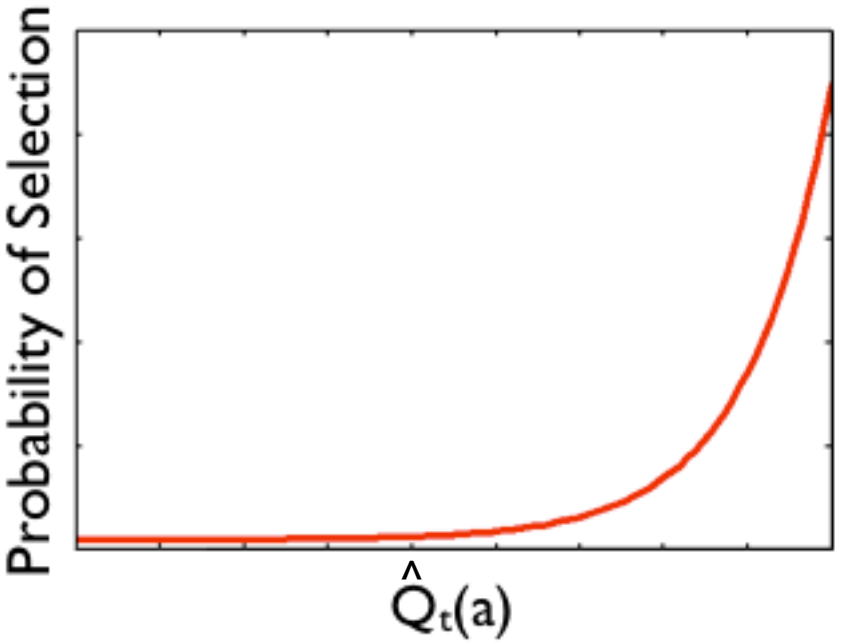


Figures: Sutton&Barto, RL:AI

Exploration strategies

Soft strategies: ensure non-0 probability of choosing any action





Choose action deemed best with probability 1-ε, random otherwise

$$p(a) = \frac{e^{Q(a)/\tau}}{\sum_{a'} e^{Q(a')/\tau}}$$

Figures: Sutton&Barto, RL:AI

Exploration strategies

Optimism:

Use mean and uncertainty (Upper confidence bound)

What happens now when optimal action wasn't tried yet?

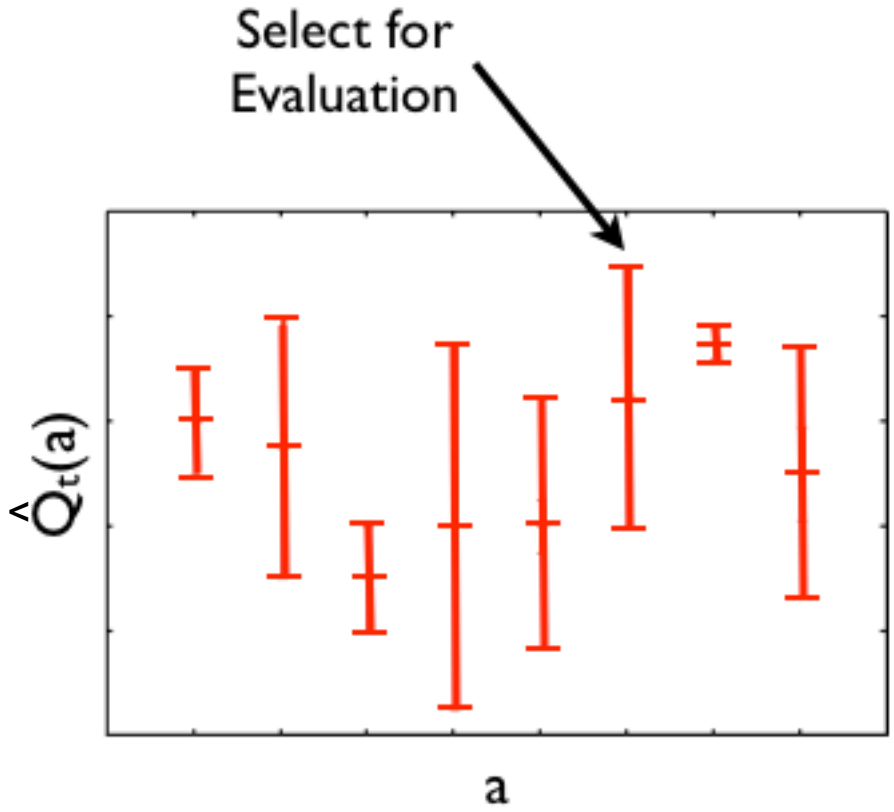


Figure: Sutton&Barto, RL:AI

Exploration strategies

Optimism:

Use mean and uncertainty (Upper confidence bound)

What happens now when optimal action wasn't tried yet?

Alternative: Initialize all Q values optimistically, then only allow them to change slowly

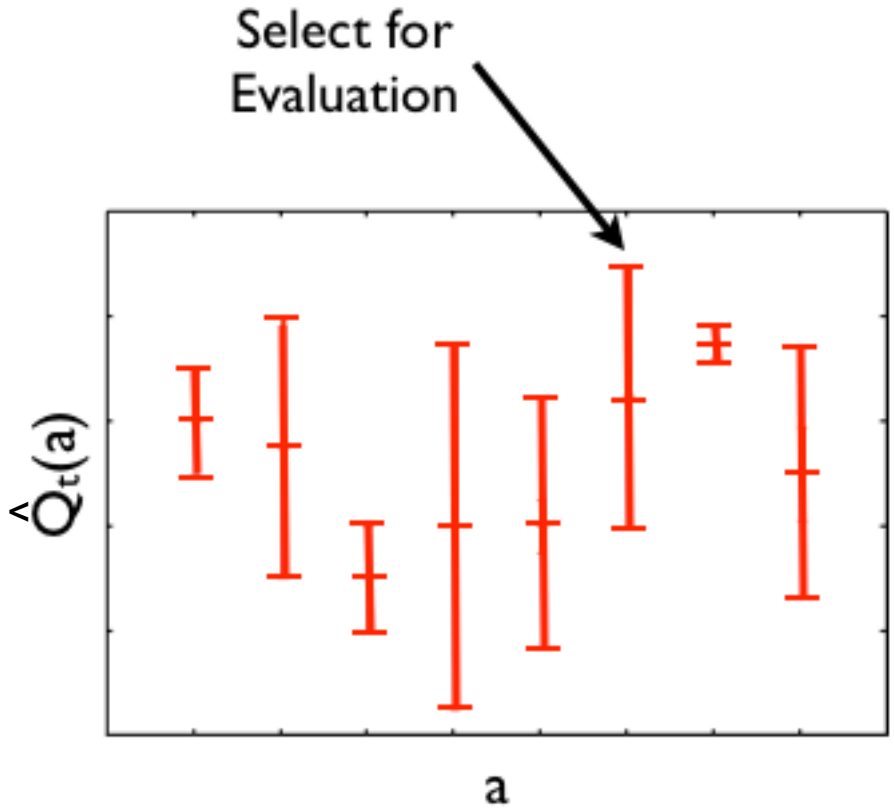


Figure: Sutton&Barto, RL:AI

Most decision problems are different…

In many decision problems, we aren't just searching for one 'optimal' action

- There are many different situation (*states* s)
- The best action depends on the situation
- The bandit framework can be extended to learn policies

$$a = \pi(s)$$

Most decision problems are different…

In many decision problems, we aren't just searching for one 'optimal' action

- There are many different situation (*states* s)
- The best action depends on the situation
- The bandit framework can be extended to learn policies

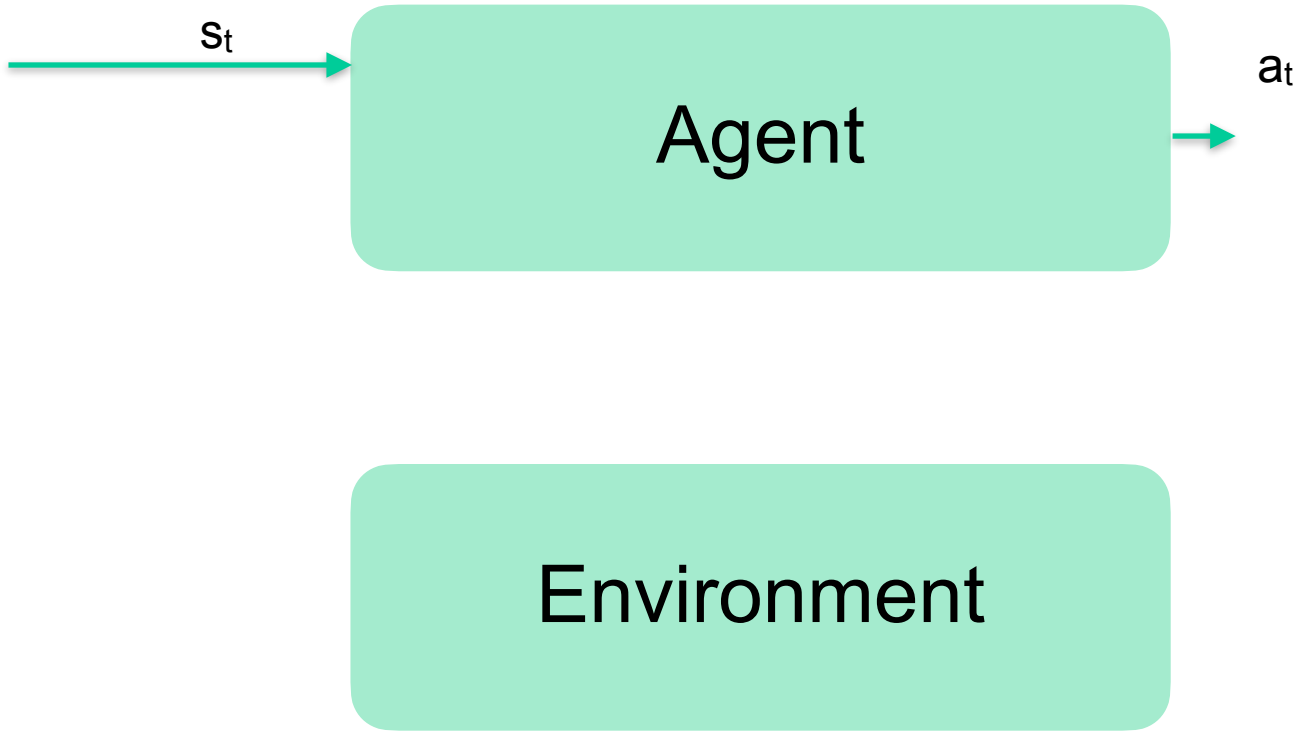
$$a = \pi(s)$$

More complex: actions can change the current situation, and affect which action should be chosen next

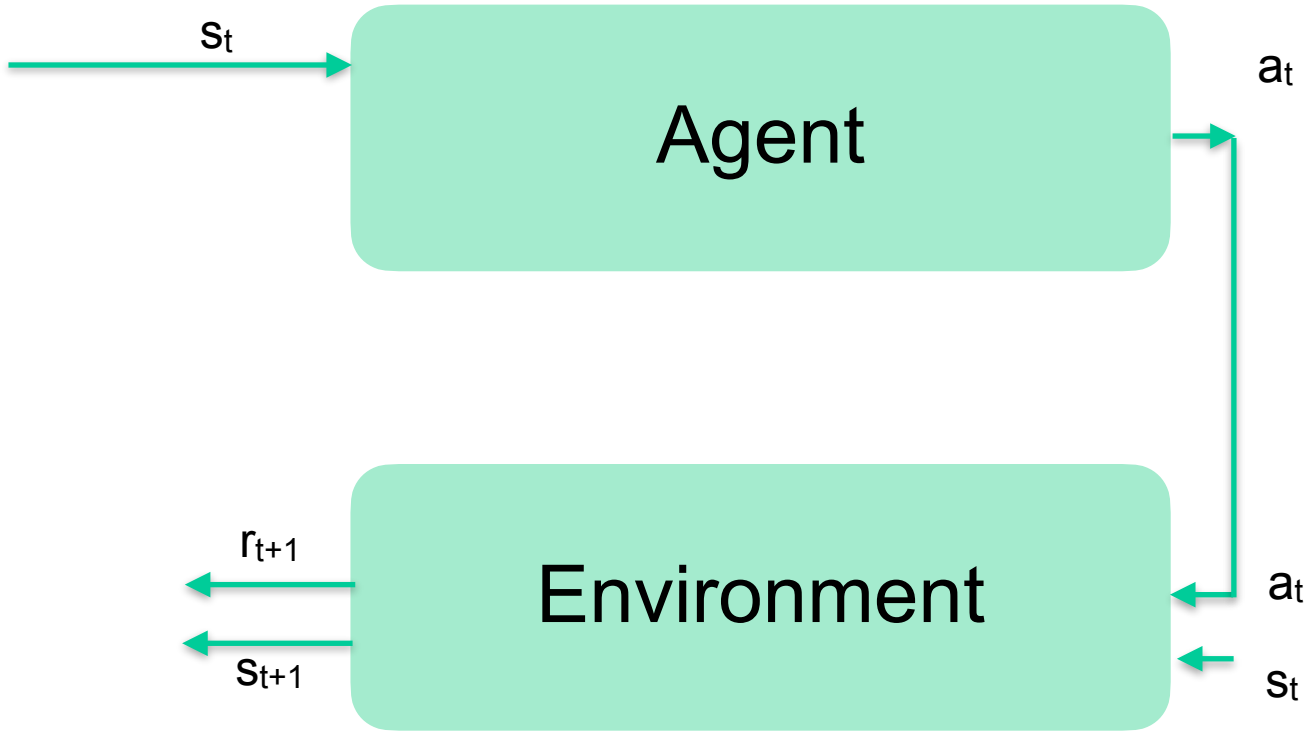
- Now we have a decision process
- Let's look at a simple model for such processes
- This will be the main topic for the rest of the course!

Markov decision process

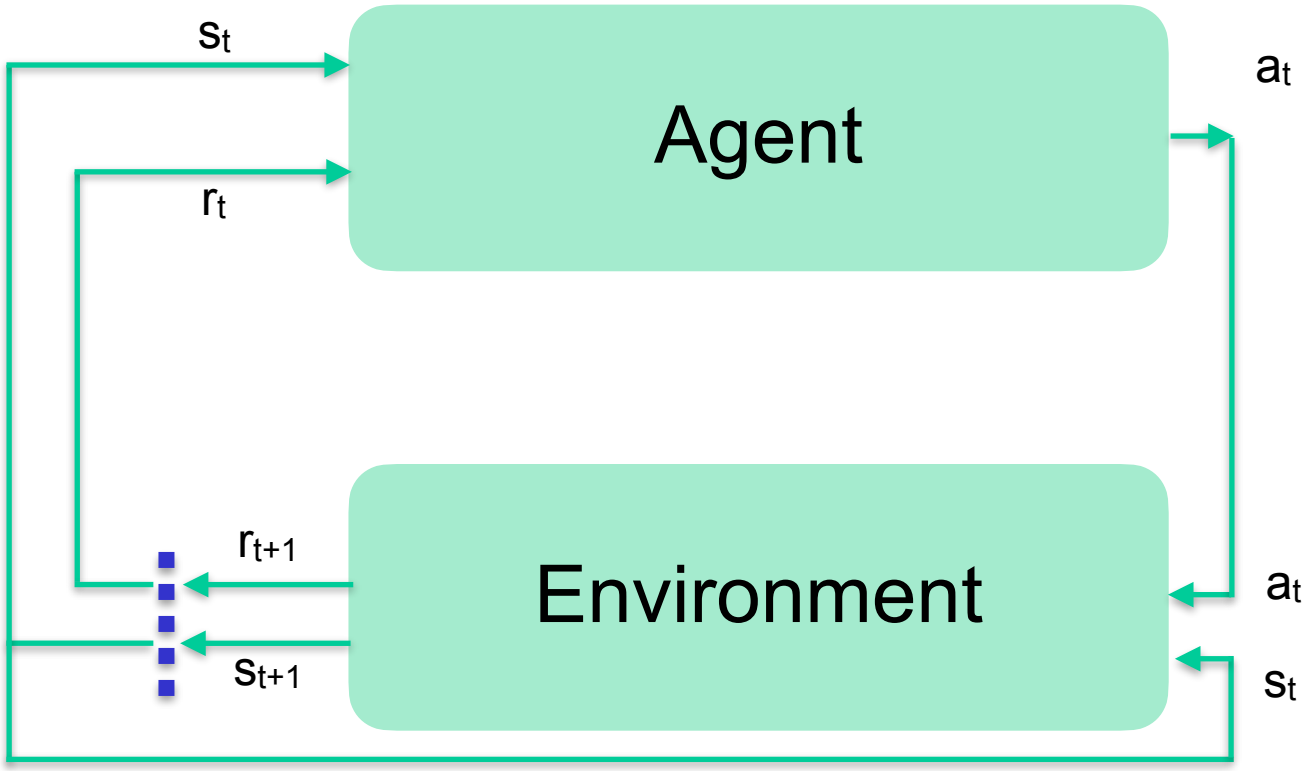
Markov decision process



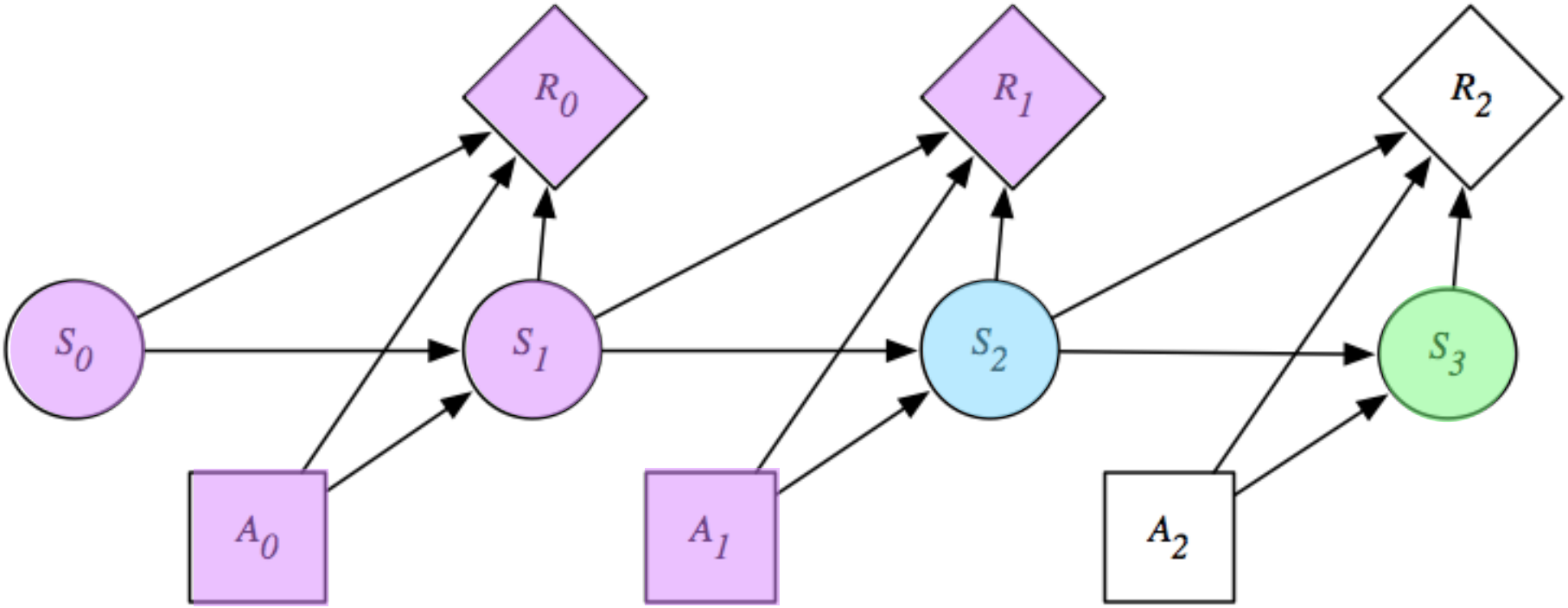
Markov decision process



Markov decision process



Markov decision process



artint.info

Markov decision processes (MDPs)

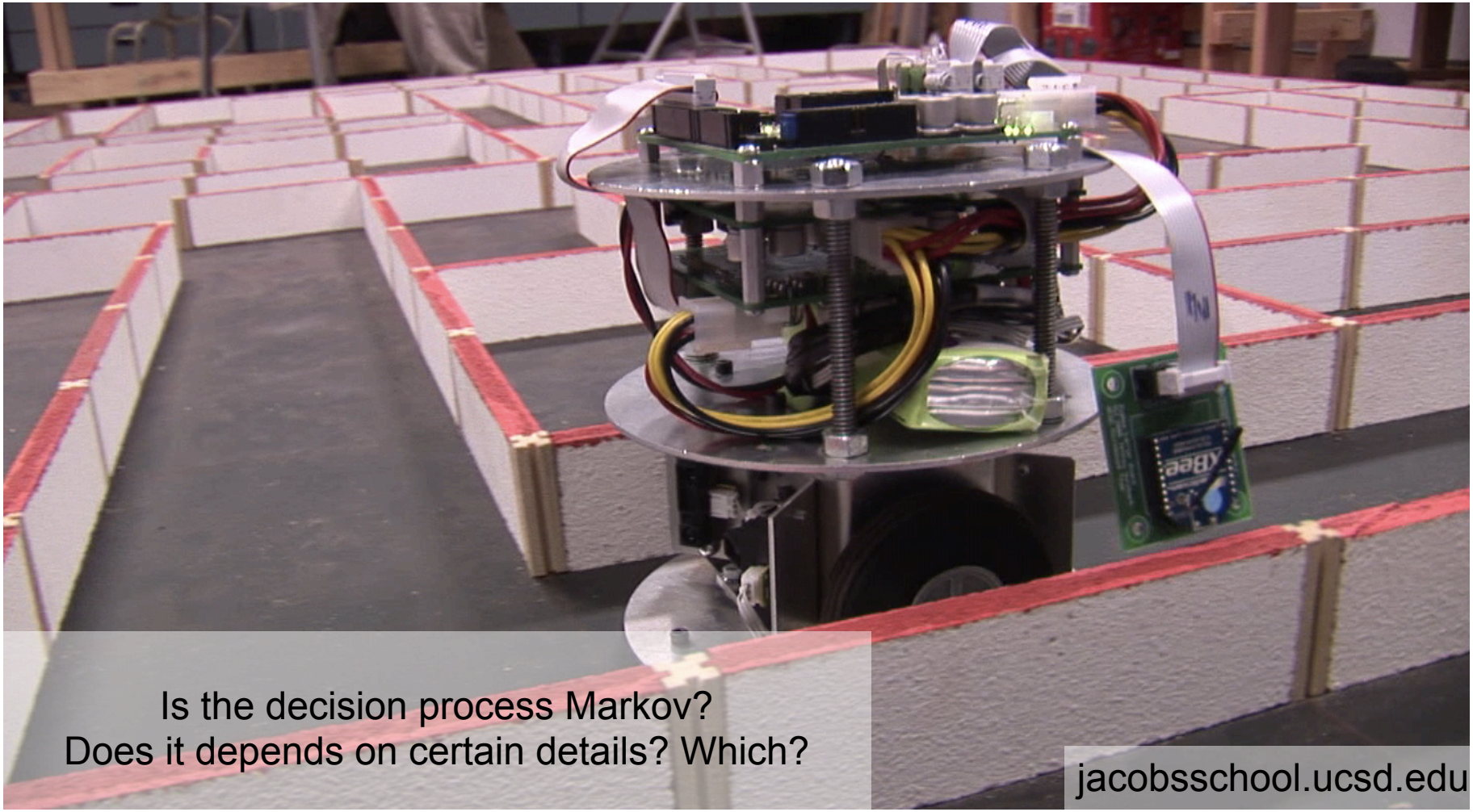
Assumptions:

- The next state depends **only** on the current state and action, and not on any states from longer ago
- Rewards depends **only** on (some of) the state, action, and next state
- Discrete time steps
- Environment is fully observable (state we are in is known and can be used for decision making). There is no hidden information.

This implies:

- The environment is **stationary**. Transitions, reward, or termination cannot depend on the time (*unless* the time is part of the state).

Markov decision process



Markov decision processes (MDPs)

Most work in RL is focused on MDPs. Why are they so hard?

- Credit assignment is hard:

  Suppose we get a good reward. Was the last action the most important in getting it? Or did an earlier action put us in a good position?

  Suppose we get a bad reward. Maybe, the last action was the best from a bad situation, and it could have been even worse…

Markov decision processes (MDPs)

Most work in RL is focused on MDPs. Why are they so hard?

- Credit assignment is hard:

  Suppose we get a good reward. Was the last action the most important in getting it? Or did an earlier action put us in a good position?

  Suppose we get a bad reward. Maybe, the last action was the best from a bad situation, and it could have been even worse…

- Data is non-iid and depends on the actions chosen

  Data comes in sequences. This violates common ML assumptions

  Initial actions might be very bad, and stay in bad states. When we learn to take better actions, suddenly we get to good states but we don't know what to do…

Markov decision processes

Formally, a finite MDP consists of:

- A finite set of states
- A finite set of actions for each state (often the same in all states)
- A dynamics function

$$p\left(s', r|s, a\right) \doteq \Pr\left\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\right\}$$

Sometimes written as: transition function $p\left(s'|s, a\right)$

reward function $p(r|s, a, s')$

- A discount factor $\gamma \in [0,1)$

Return and horizons

Rewards encode what to achieve

We want to maximise the expected cumulative reward

Cumulative reward is called the **return**

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$$

where T is length of episode (one game, one run in maze, …)

This only makes sense if the task is **episodic**
(every episode ends in a special *terminal state*)

Return and horizons

In **continuing tasks** the return is problematic!

We'll use the notion of **discounted return**

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Rest of the course: finding optimal policies

Next lecture:
define optimal policies & find them for known MDPs

After that:

learn policies for **unknown** MDPs

Various conditions, e.g. discrete / continuous states and actions

Thanks for your attention!

Feedback?

h.c.vanhoof@uva.nl