(Partially Observable) Markov Decision Processes

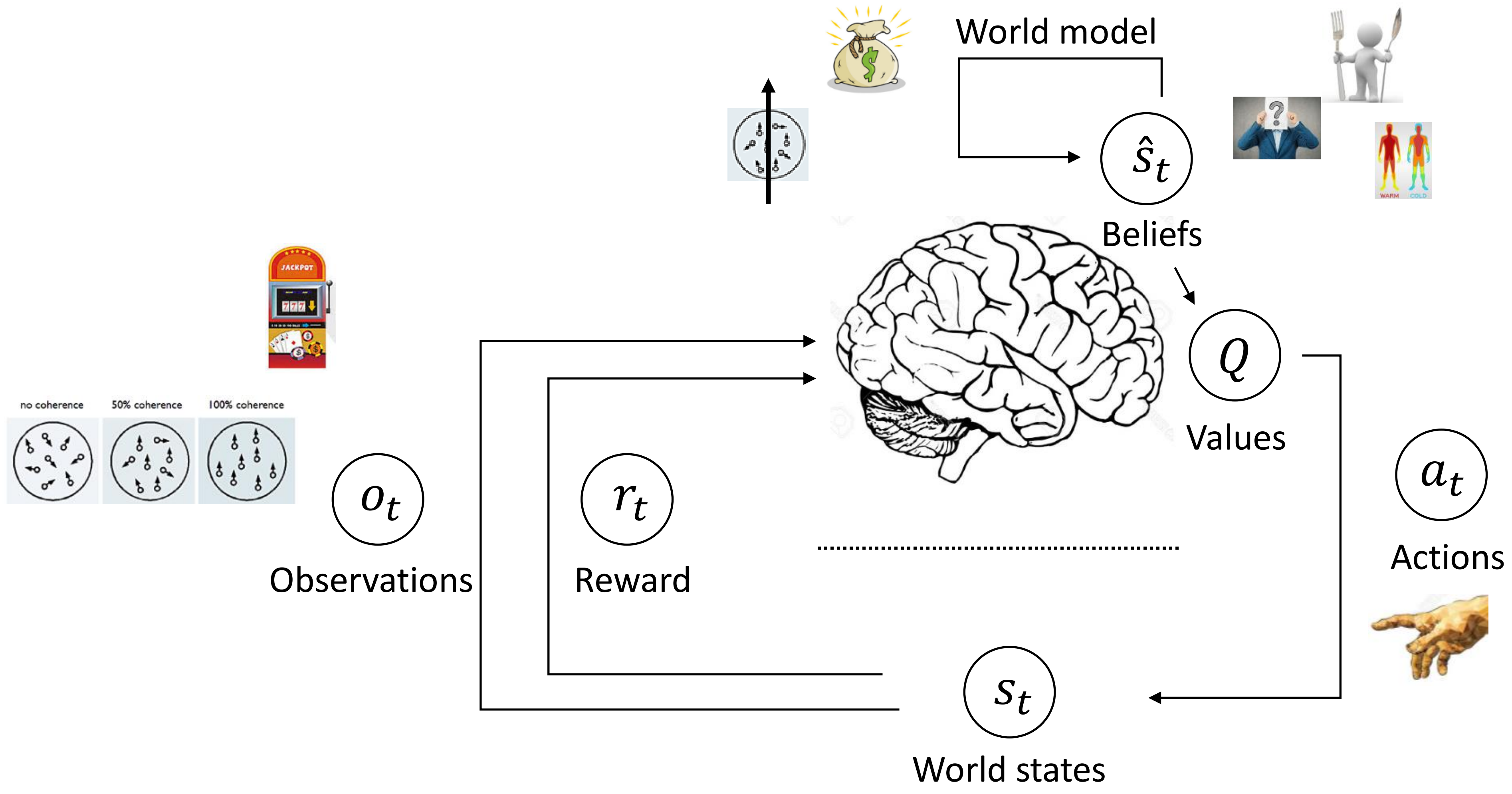
Philipp Schwartenbeck

Wellcome Trust Centre for Human Neuroimaging, UCL

Computational Psychiatry Course 2020, TNU, Zurich

Why (PO)MDPs?

If we **build a model of the world** we can use it for sophisticated **learning** and **inferences**



- It's a good idea to play the slot machine *if* you think $P(win)$ is high
- It's a good idea to look for food *if* you are hungry
- It's a good idea to gain information *if* you are uncertain

POMDPs in action

How did you decide whether to attend a talk at CPC?

- Let's assume you had to choose between two options


Attend a talk

Go for a swim with your friends

It's very rewarding to attend a talk that is interesting, but more rewarding to go for a swim if the talk is less interesting

How do you decide?

- If you are uncertain about how interesting the talk will be (hidden state), it might be a good idea to check (gather information)
- Checking might be costly (if you wait too long you will miss the beginning of the talk)

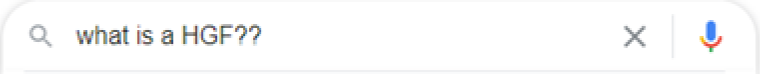
Highlights important issues

- Interplay of state inference and reward maximisation
- Value of information
- Different aspects that can go wrong – relevance for comp psych!

SCHEDULE
CPCZurich2020

PSYCHIATRY

MONDAY – 7th September 2020		
08:15 – 08:40	Introduction to Day 1	Klaas Enno Stephan
	• Videoconferencing Platform	Frederike Petzschner
	• Introduction to the CPC Course	Katharina Wellstein
08:40 – 09:30	Introduction to Computational Psychiatry	Klaas Enno Stephan
09:30 – 10:20	Schizophrenia	Jakob Siemerkus
10:20 – 10:50	Break	
10:50 – 11:40	Affective Disorders	Gina Paolini
11:40 – 12:30	Bipolar Disorder	Helen Schmidt
12:30 – 14:00	Lunch Break	
14:00 – 14:50	Psychosomatics	Roland von Känel
14:50 – 15:40	Addiction	Marcus Herdener
15:40 – 16:10	Break	
16:10 – 17:00	Introduction to Computational Modeling	Klaas Enno Stephan



[You should attend every talk at the CPC]

Structure and Outline

Where are we: *Day 3 MODELS OF ACTION SELECTION*

Reinforcement Learning $\xrightarrow{\text{RL II: model based RL}}$ (PO)MDPs $\xleftarrow[\text{Probability theory (task dynamics)}]{\text{Bayesian inference}}$

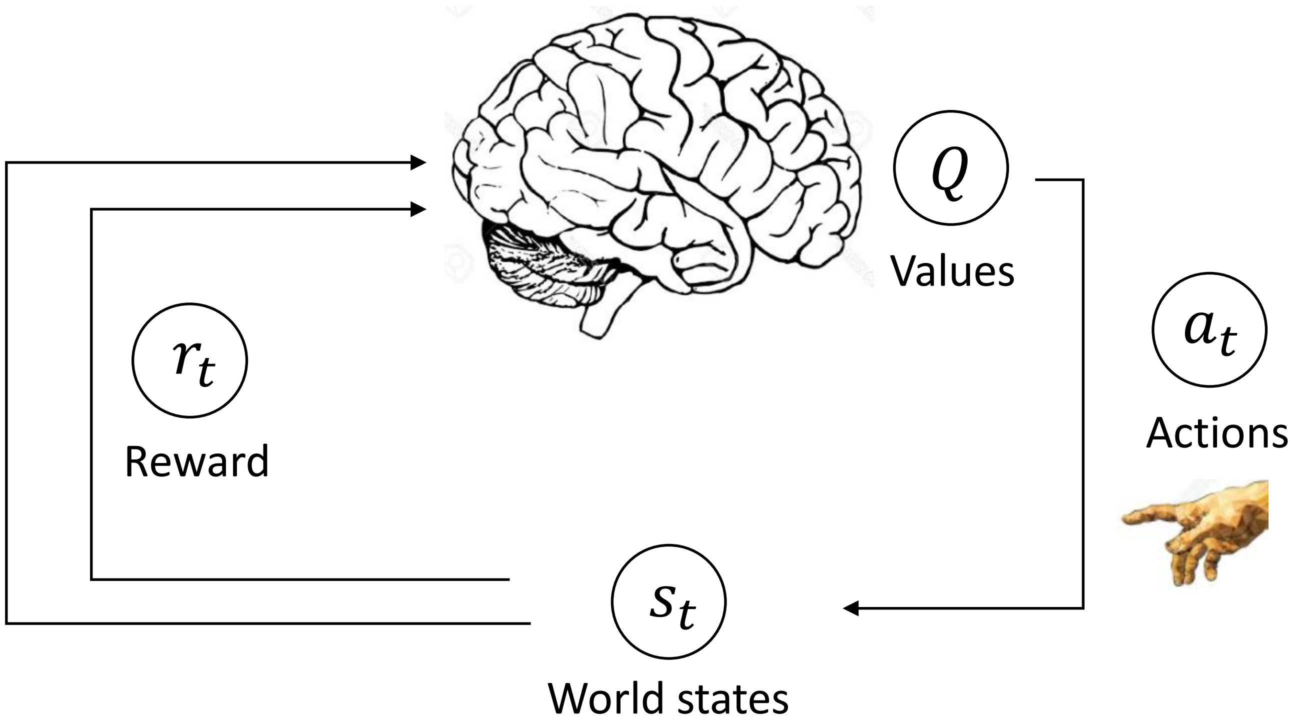
Models of Perception: Psychophysics
Models of Perception: Bayesian models
Models of Perception: Predictive Coding
From Perception to Action: HGF
Models of Action Selection: Active Inference

<u>Outline</u>

I. Markov Decision Processes (MDPs)
II. Solving MDPs
III. Belief-based MDPs (POMDPs)
IV. [Current Research]

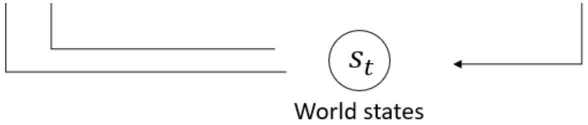
I. Markov Decision Processes (MDPs)

Markov Decision Processes

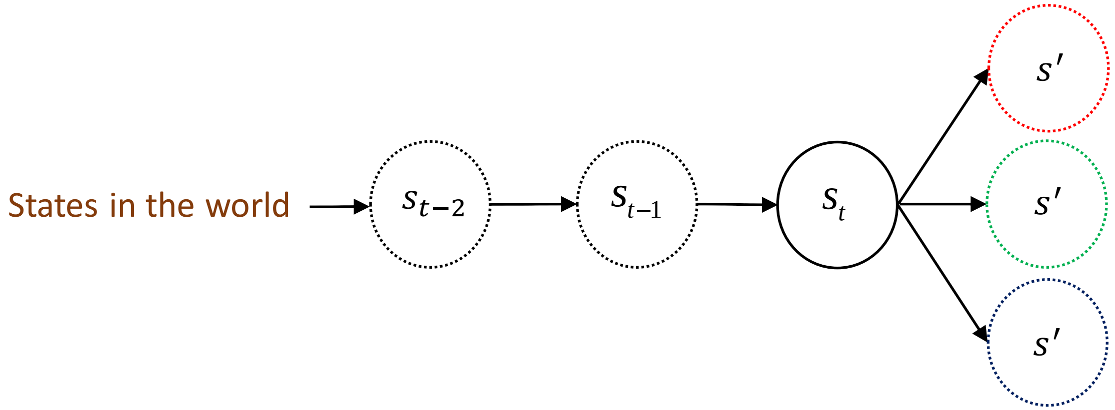


Markov Process → **Markov Reward Process** (MRP) → **Markov Decision Process** (MDP)

Markov Processes



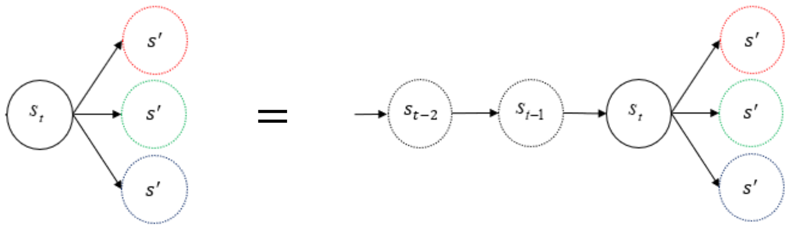
We are dealing with problems that we can model as a **sequence of discrete states**:



$s' \stackrel{\text{def}}{=}$ next state

Fundamental assumption: **Markov property**

- When predicting the next state, we only need the current state
- “The future is independent of the past given the present”

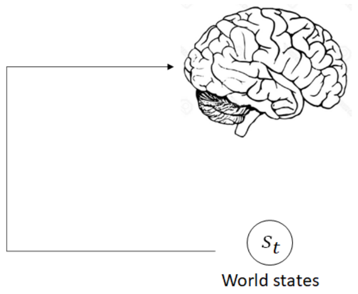


Formally: State s_t is **Markov** if and only if $P(s'|s_t) = P(s'|s_t, s_{t-1}, \dots, s_1)$

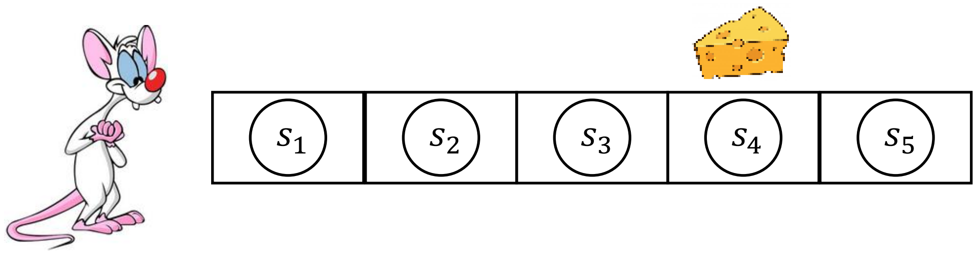
- We are *equally good* at predicting the next state if we *only* take the current state or if we take *all* the previous history

Q: Can you think of situations where this doesn’t work?

Markov Processes

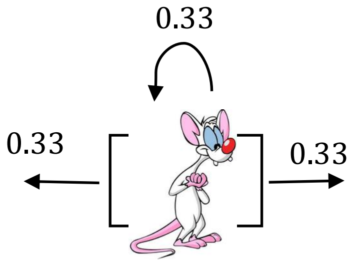


Let's assume
- There is a mouse on a linear track
- There is a reward in one state on that track (s_4)
- Random starting position

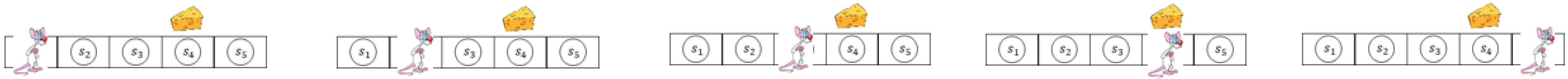


In any state, the mouse can move **left** or **right** or **stay** where it is.

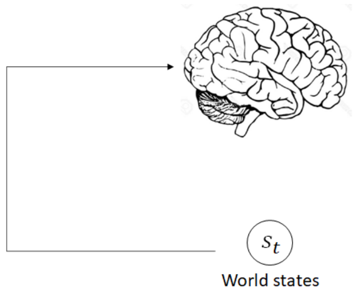
However, it turns out our mouse isn't very clever
- It has no concept of rewards
- It randomly moves around – *no active control*



This simple process allows us to define a **state space** $\boldsymbol{S}$:

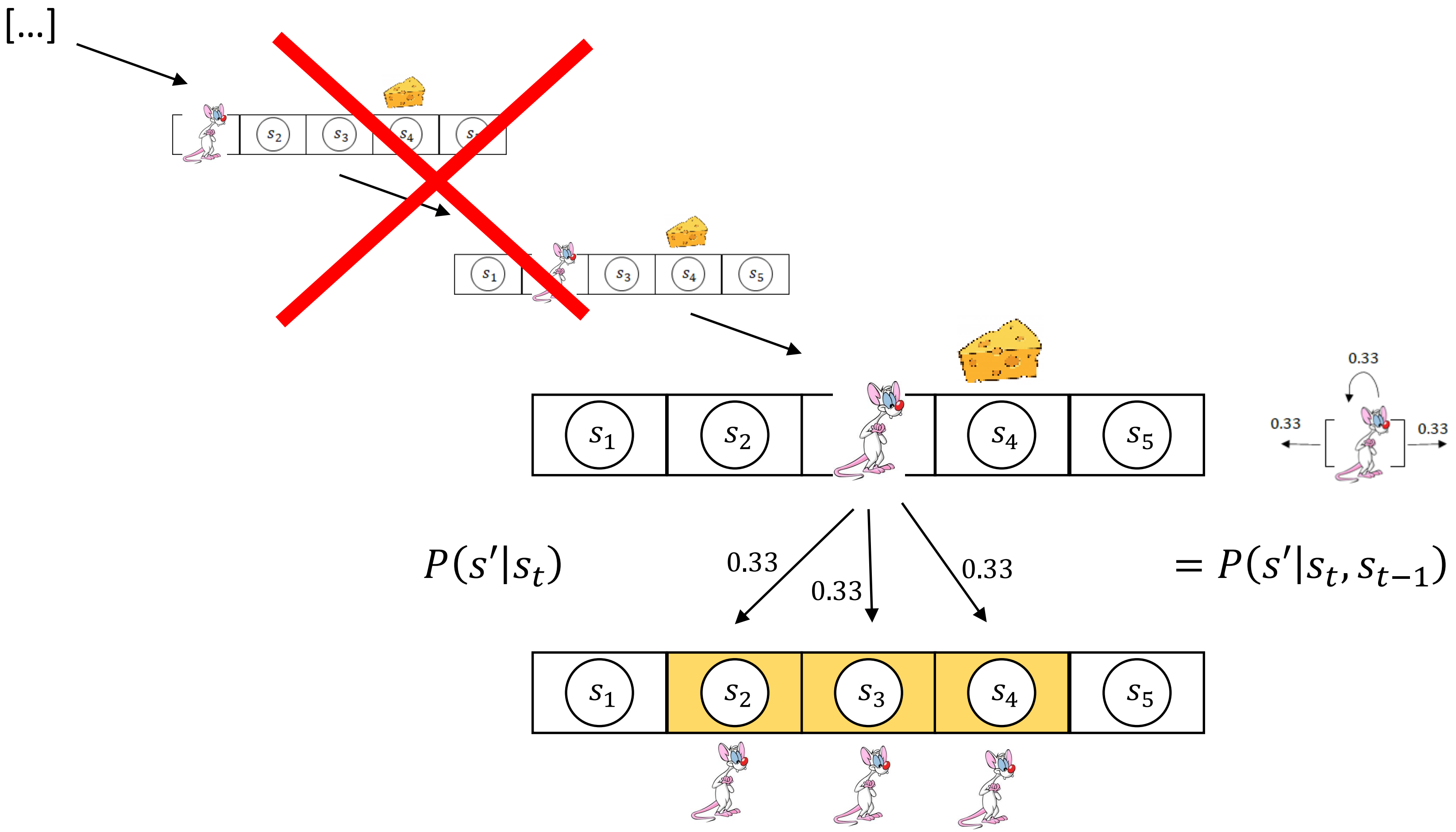


Markov Processes



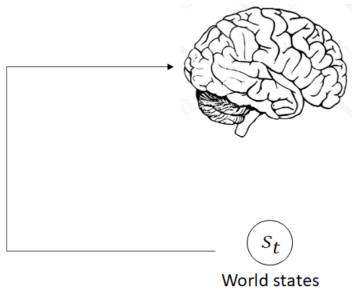
This satisfies the **Markov Property**!

Let's consider the probability of a next state:



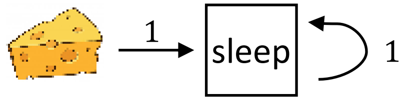
$$P(s'|s_t) = P(s'|s_t, s_{t-1}) = P(s'|s_t, s_{t-1}, s_{t-2}) = [\ldots]$$

Markov Processes



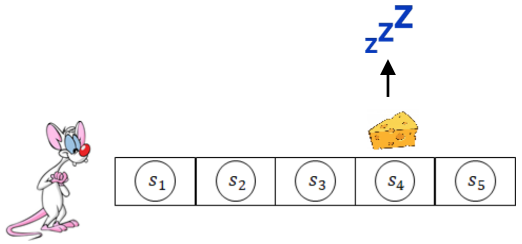
Additional simplifications:

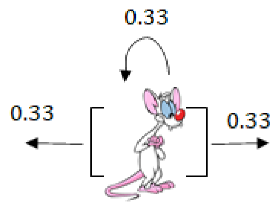
- Often, we also define **terminal (absorbing) states**, such as reaching a goal
- Moving left in leftmost state => stay (same for rightmost state)



This allows us to define **transition probabilities P**:

$$P = \textit{from}\overset{\textit{to}}{\begin{bmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{n1} & \cdots & P_{nn} \end{bmatrix}} = \begin{array}{cc} & \begin{array}{cccccc} s_1 & s_2 & s_3 & s_4 & s_5 & \text{sleep} \end{array} \\ \begin{array}{c} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ \text{sleep} \end{array} & \begin{bmatrix} 0.66 & 0.33 & 0 & 0 & 0 & 0 \\ 0.33 & 0.33 & 0.33 & 0 & 0 & 0 \\ 0 & 0.33 & 0.33 & 0.33 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0.33 & 0.66 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

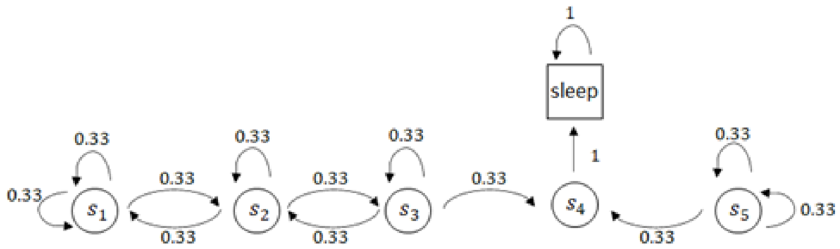




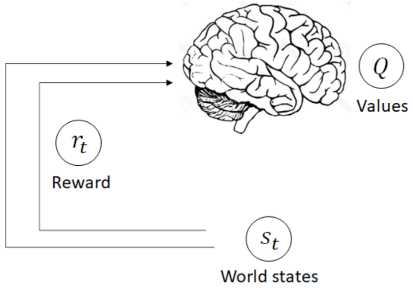
This allows us to define a **Markov Process** or **Markov Chain** (S,P) based on:

- A (finite) state space S
- Transition probabilities $P(\mathrm{S}_{\mathrm{t}+1} = \mathrm{s}' | \mathrm{S}_{\mathrm{t}} = \mathrm{s})$

Can define the following **transition graph** for our problem:

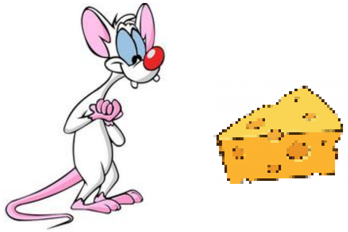


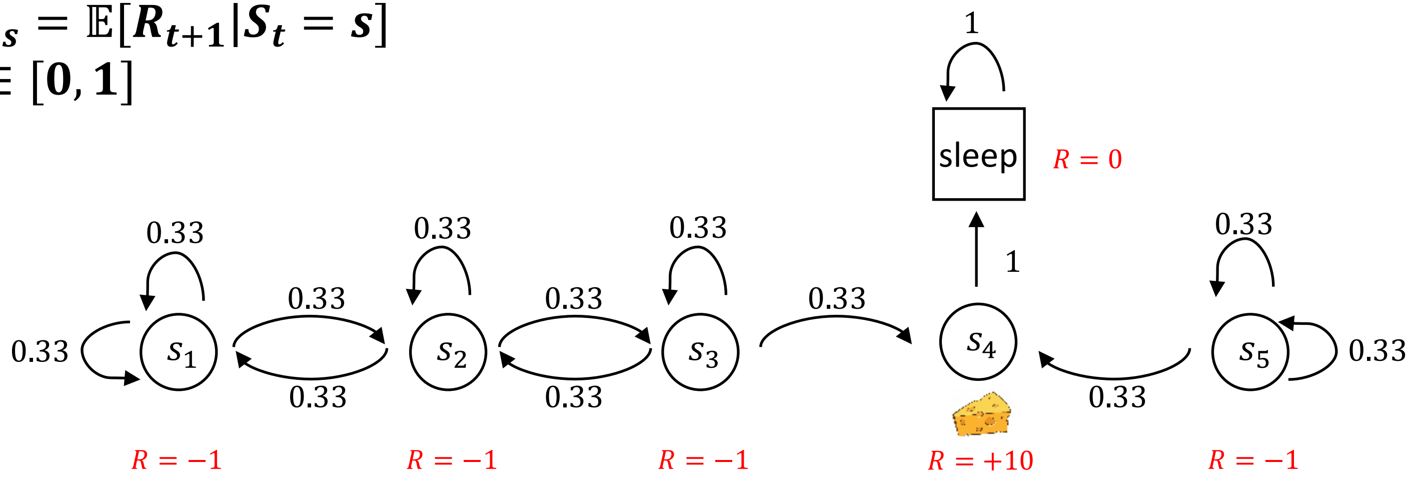
Markov Reward Processes



We can add *rewards* to our Markov Process to define a **Markov Reward Process** (S,P,R,γ)

- A (finite) state space S
- Transition probabilities $P(\mathrm{S}_{t+1} = \mathrm{s}'|\mathrm{S}_t = \mathrm{s})$
- **Reward function** $\boldsymbol{R_s} = \mathbb{E}[\boldsymbol{R_{t+1}}|\boldsymbol{S_t} = \boldsymbol{s}]$
- **Discount factor** $\boldsymbol{\gamma} \in [\mathbf{0}, \mathbf{1}]$





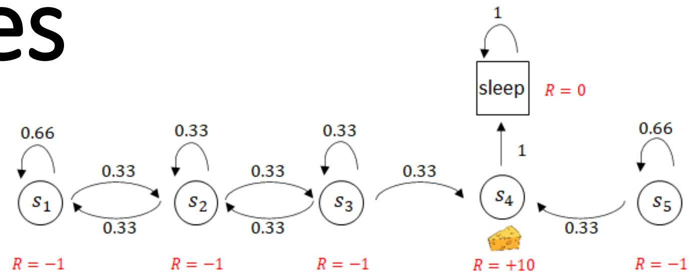
Allows us to define the '**return**':

- γ determines how much we take the future into account

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

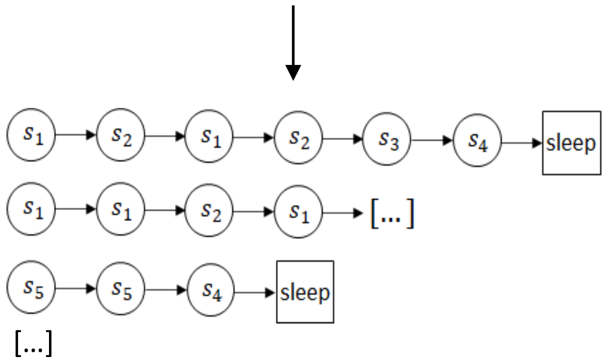
Allows us to define the '**expected return**' of a state, i.e. the **state-value function**: $v(s) = \mathbb{E}[G_t|s]$

Markov Reward Processes



We can now express the transition graph in terms of the **value of states** $v(s) = \mathbb{E}[G_t|s]$

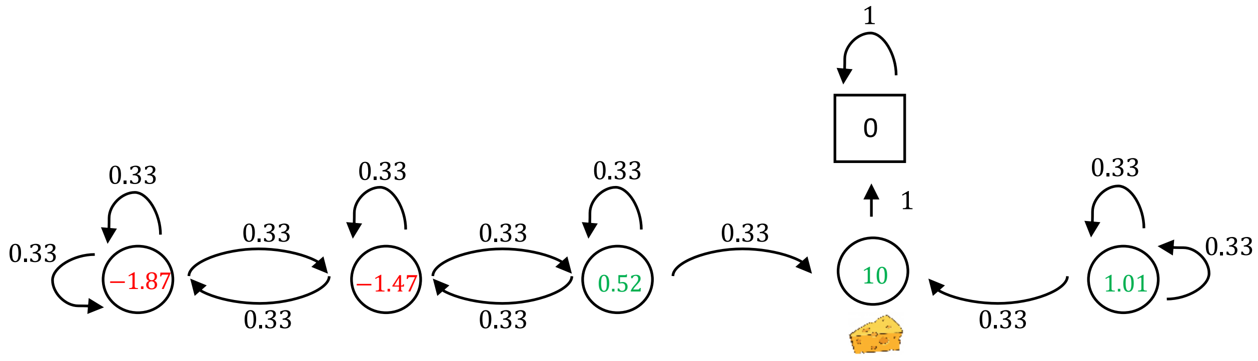
- i.e the **expected future return** starting from a state $s \in \{s_1, s_2, s_3, s_4, s_5\}$



How do we find the value of a state? Simplest approach:

- **Randomly sample sequences** from a **random starting state** and **compute return**
- Then average

$\boldsymbol{\gamma = 0.5}$:



$-1 + 0.5 * (-1) + 0.5^2 * (-1) + 0.5^3 * (-1) + 0.5^4 * (-1) + 0.5^5 * (10) + 0.5^6 * (0)$

$-1 + 0.5 * (-1) + 0.5^2 * (-1) + 0.5^3 * (-1) + \cdots$

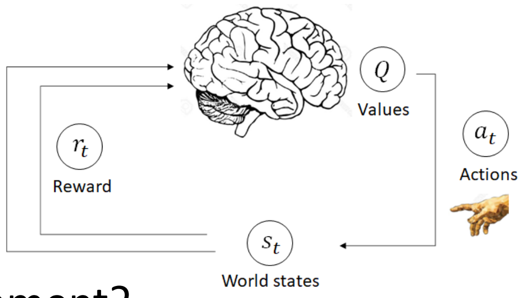
$-1 + 0.5 * (-1) + 0.5^2 * (10) + 0.5^3 * (0)$

[...]

More of that in a bit..

Q: Do you think the brain is doing something like this?

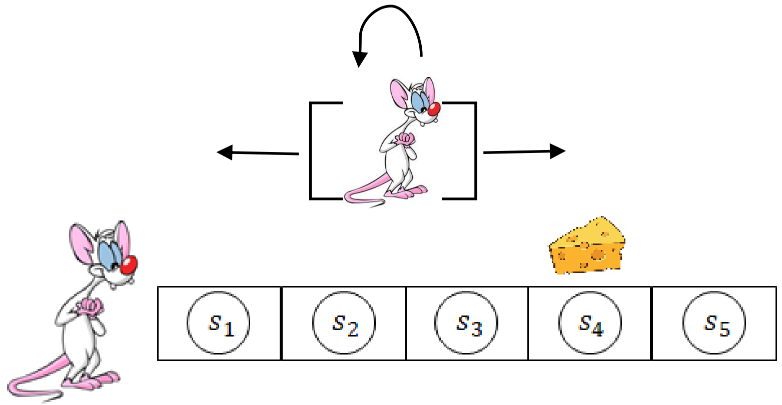
Markov Decision Processes



So far we've assumed random movement - what about agents who can actively control their environment?

We can add *actions* to our Markov Reward Process to define a **Markov Decision Process** (S,P,R,γ,A)

- A (finite) state space S
- **Finite set of actions A**
- Transition probabilities $P(\mathrm{S}_{\mathrm{t}+1} = \mathrm{s}' | \mathrm{S}_{\mathrm{t}} = \mathrm{s}, \mathbf{A_t} = \mathbf{a})$
- Reward function $R_s = \mathbb{E}[R_{t+1} | S_t = s, \boldsymbol{A_t} = \boldsymbol{a}]$
- Discount factor $\gamma \in [0,1]$



Movement not stochastic process any more, but actively controlled via **policies**

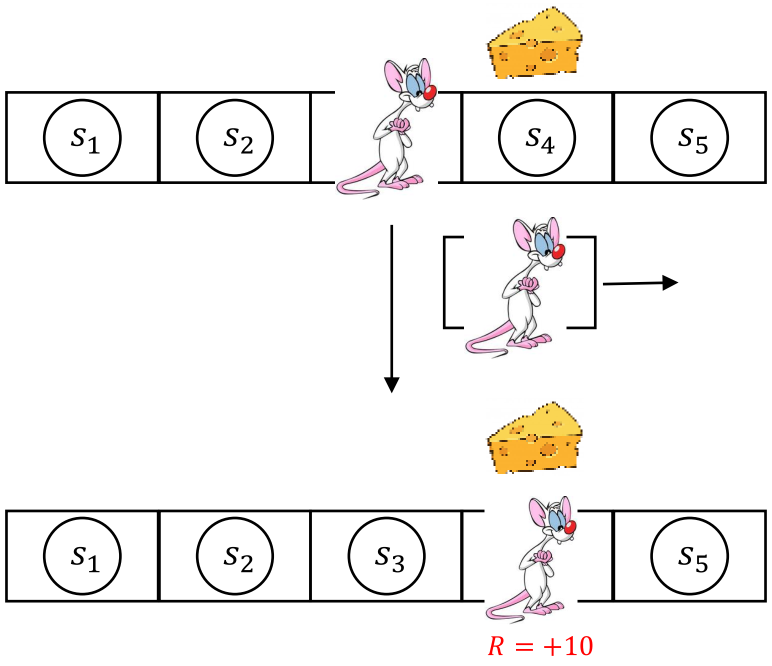
A **policy** is a distribution over actions given a state: $\pi(a|s) = P(A_t = a | S_t = s)$

- Often not deterministic (cf., exploration)

MDPs provide the basis for 'model-based decision-making'

Fundamental equation: $P(s', r|s, a) = P(S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a)$

Specifies all environment dynamics!

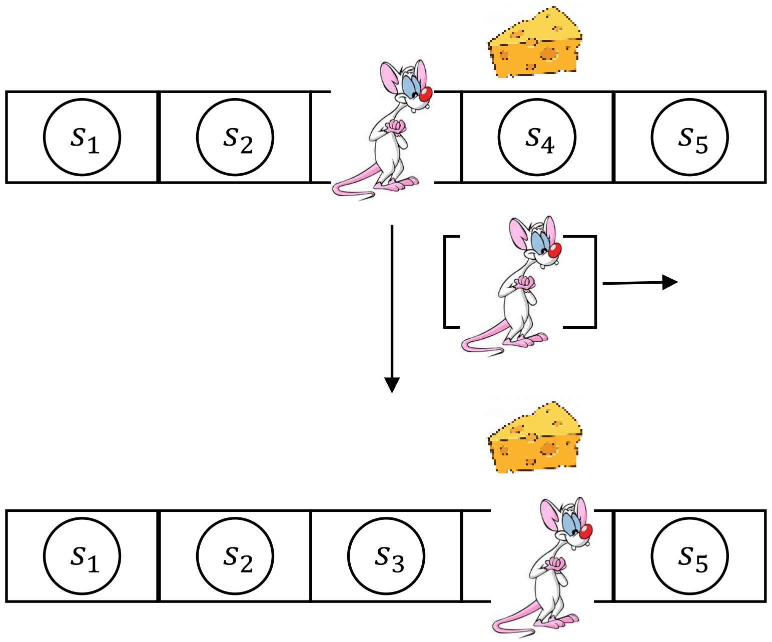


MDPs provide the basis for 'model-based decision-making'

$$P(s',r|s,a) = P(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a)$$

Encodes anything we might want to know about environment, such as **state-transition probabilities**:

$$P(s'|s,a) = P(S_{t+1} = s' | S_t = s, A_t = a) = \sum_r P(s',r|s,a)$$



to

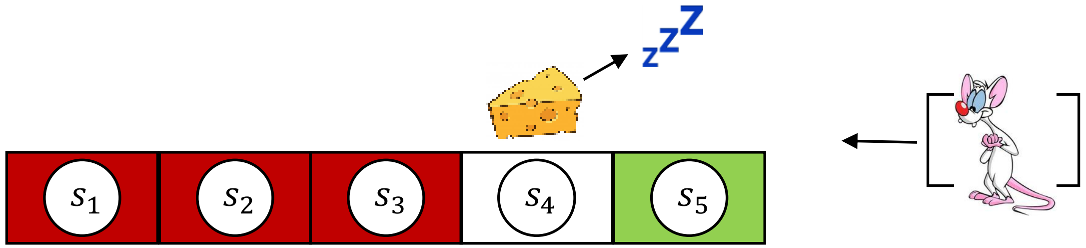
$$\text{e.g.}\quad P(s'|s,right) = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ \text{sleep} \end{array} \begin{array}{c} \begin{array}{cccccc} s_1 & s_2 & s_3 & s_4 & s_5 & \text{sleep} \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \quad \text{from}$$

These transition probabilities depend on action!

MDPs provide the basis for 'model-based decision-making'

Can also encode **expected reward for state-action pair**
– which action is particularly good in which state?

$$r(s,a) = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right] = \sum_r r \sum_{s'} P(s', r \mid s, a)$$

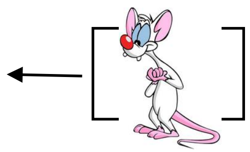


Can also encode **expected reward for state-action-next state triplet**
– which state-to-state transition is particularly good for a given action?

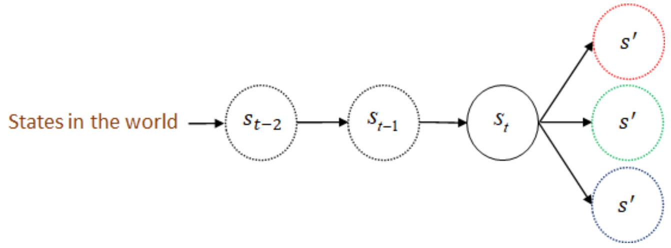
$$r(s,a,s') = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s'\right] = \sum_r r \frac{P(s', r \mid s, a)}{P(s' \mid s, a)}$$

to

$$r(s,a,s') = \begin{array}{c|cccccc|l} & s_1 & s_2 & s_3 & s_4 & s_5 & \text{sleep} & \\ \hline s_1 & -1 & 0 & 0 & 0 & 0 & 0 & \\ s_2 & -1 & 0 & 0 & 0 & 0 & 0 & \\ s_3 & 0 & -1 & 0 & 0 & 0 & 0 & \text{from} \\ s_4 & 0 & 0 & 0 & 0 & 0 & 0 & \\ s_5 & 0 & 0 & 0 & 10 & 0 & 0 & \\ \text{sleep} & 0 & 0 & 0 & 0 & 0 & 0 & \end{array}$$



Markov Decision Processes interim summary



MDPs are **actively controlled Markov Processes**

- **Markov process** models evolution of state based on **Markov property** (current state summarises history)
- **Markov Reward Process** introduces concept of reward – allows to define value of states
- **Markov Decision Process** enables **active** (**optimal**) **control**

Allows us to model and provide full account of **environment dynamics**

- Fundamental for *model-based* decision-making

Provides basis for **optimal control** problems

- Substantial contribution to our understanding of (neural) computations underlying decision-making

Q: Is the MDP framework adequate for representing all goal-directed learning tasks? Can you think of exceptions?

II. Solving MDPs

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)."

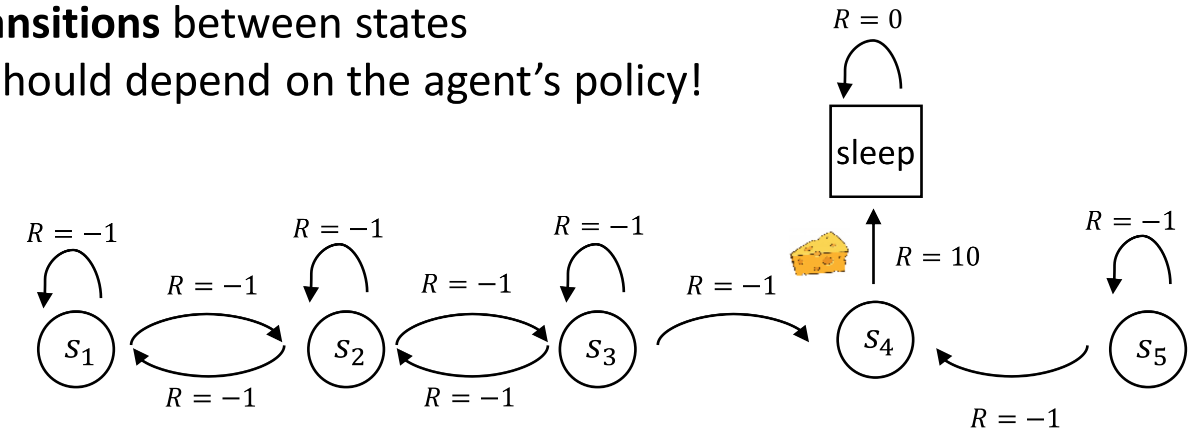
Q: Do you agree?

Solving MDPs: Transform return into value function

Let's try to maximise returns $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$

Importantly, we care about the reward of taking a **particular action** in a **particular state**

- Rewards **embedded in transitions** between states
- Thus, the value of states should depend on the agent's policy!



We can now define the value of a state as the '**expected return**' **under a given policy** $\boldsymbol{\pi}$

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

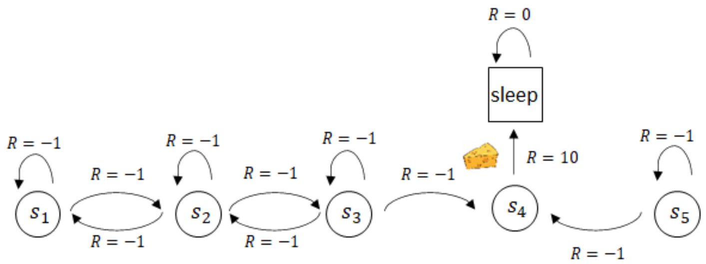
How good is a state under a given policy?

Very useful for finding good actions: **value of action a in state s under policy** $\boldsymbol{\pi}$

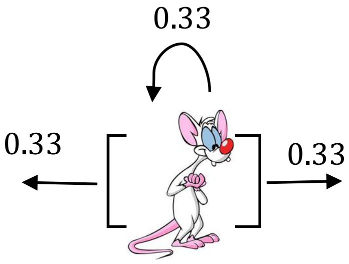
$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

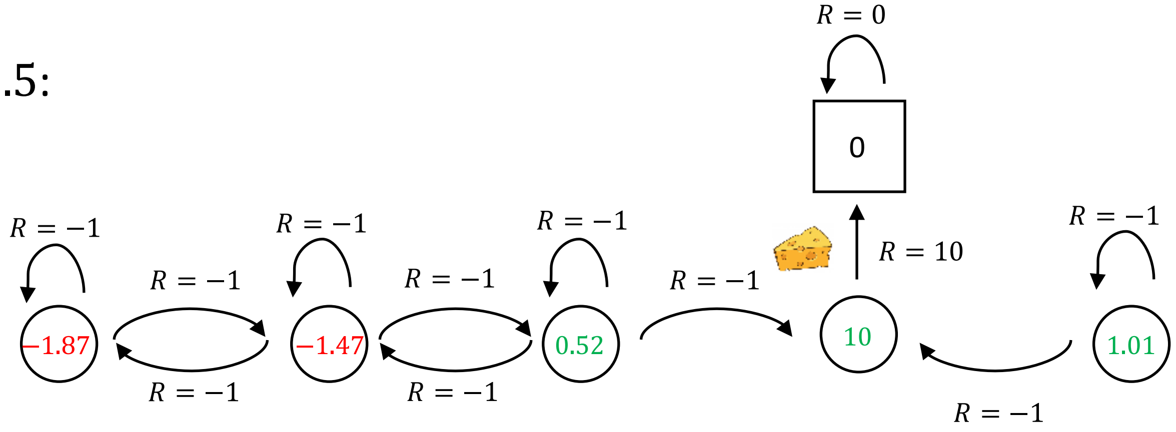
How good is an action in a particular state under a given policy?

Solving MDPs: Transform return into value function



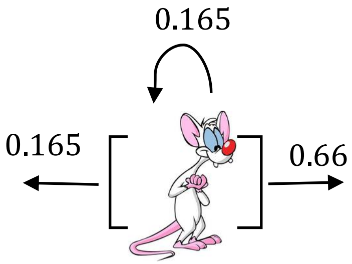
$v_\pi(s)$ for random policy and $\gamma = 0.5$:

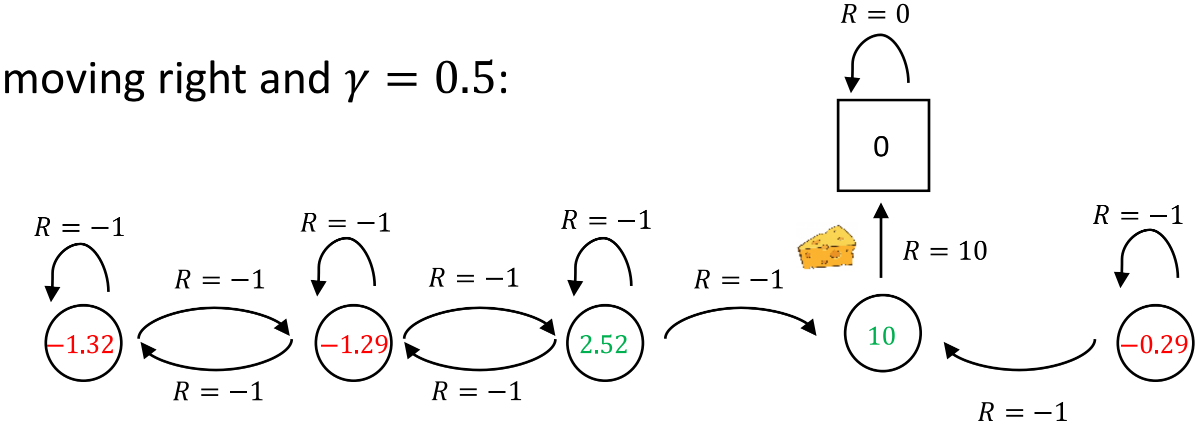




Highlights the effect of the policy on state values!

$v_\pi(s)$ for a policy biased towards moving right and $\gamma = 0.5$:





Q: What's all this about??

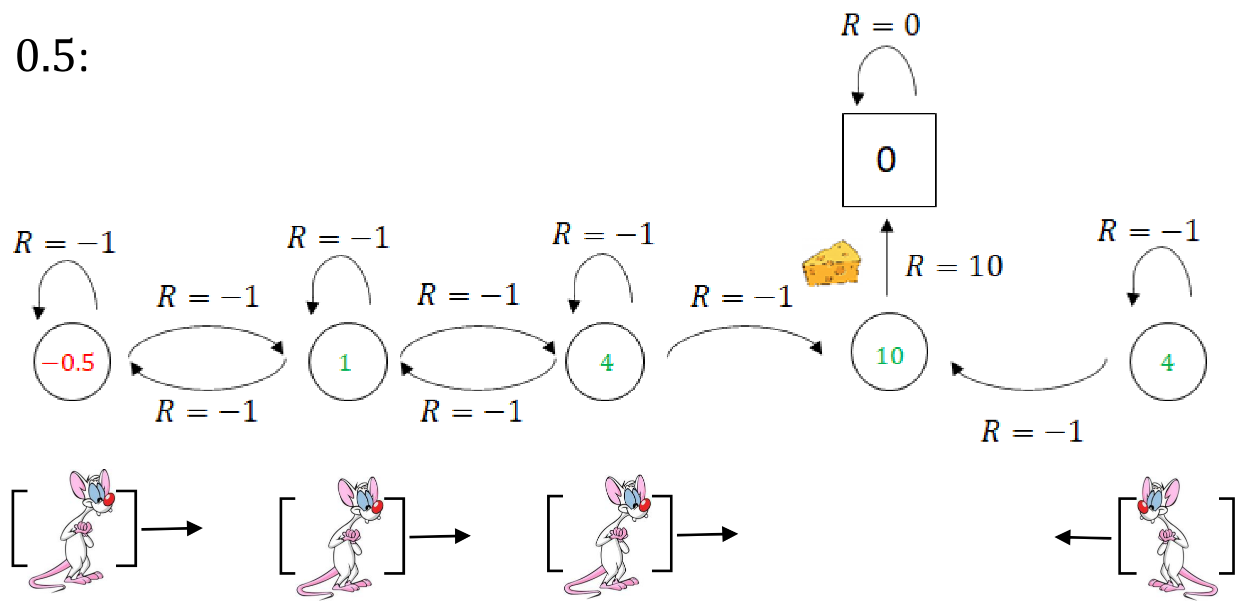
A: We can use this to look for the best policy that maximises reward!!

Solving MDPs optimally

We want the best policy, not just any policy

- **Optimal state value function**: $v_*(s) = \max_\pi v_\pi(s)$
- For every MDP, there is such an optimal policy – but it's not trivial to find it

E.g., $v_*(s) = \max_\pi v_\pi(s)$ for $\gamma = 0.5$:



Optimal value function = best possible performance MDP

Obvious here – but computationally very demanding in more realistic examples

Solving MDPs <u>optimally:</u> Bellman equation

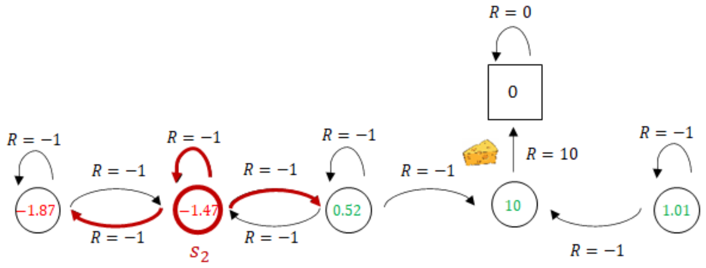
To simplify that problem, we can now do something fun:

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

Recursive definition of return

Recursive definition of value of state under given policy – **Bellman equation** *(Bellman, 1957)*

$$v_\pi(s) = \mathbb{E}_\pi[G_t | s_t]$$
$$= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | s_t]$$
$$= \sum \pi(a|s) \sum P(s', r | s, a)[r + \gamma v_\pi(s')]$$

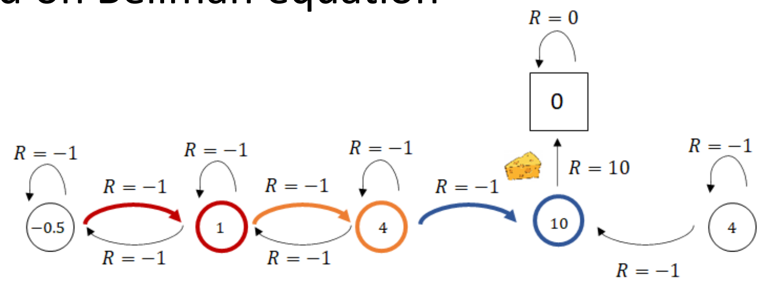


Key insight: you only need to **look one time step** ahead when computing the value!

Solving MDPs <u>optimally</u>

Once we have v_*, it is straightforward to find the **optimal action in every state** based on Bellman equation

Most straightforward way: $$q_*(s,a) = \sum P(s',r|s,a)[r + \gamma v_*(s')]$$



Only need to look one step ahead to find the optimal action!

Outlined approach for finding v_* rarely useful – akin to exhaustive search

- Look ahead at all possibilities, compute probabilities and desirability (expected reward) - Computationally demanding
- Relies on accurate representation of environment dynamics

A lot of different approximation/iterative solution methods *(See Stutton & Barto, 2018, chapters 2-6)*

- **Dynamic Programming** (roughly: clever ways to apply Bellman when going through all states and actions/policies)
  - Value iteration
  - Policy Iteration
- **Q-learning** (Watkins, 1989): off-policy iterative method
- **Sarsa**: on-policy iterative method

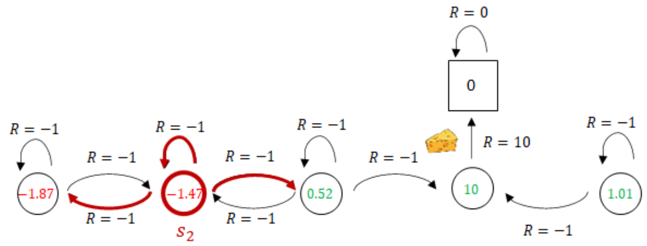
Solving MDPs: Summary

Under the reward hypothesis, we are trying to find **policies** that **maximise the expected return**

- Having built our MDP (specifying all environment dynamics), we can now **solve this to find the best actions**

In this process **Bellman equations** are really central - Important to know about it ☺

- Bellman equation/optimality based on **one-step lookahead**
- A lot of research within MDPs on how to approximate efficiently

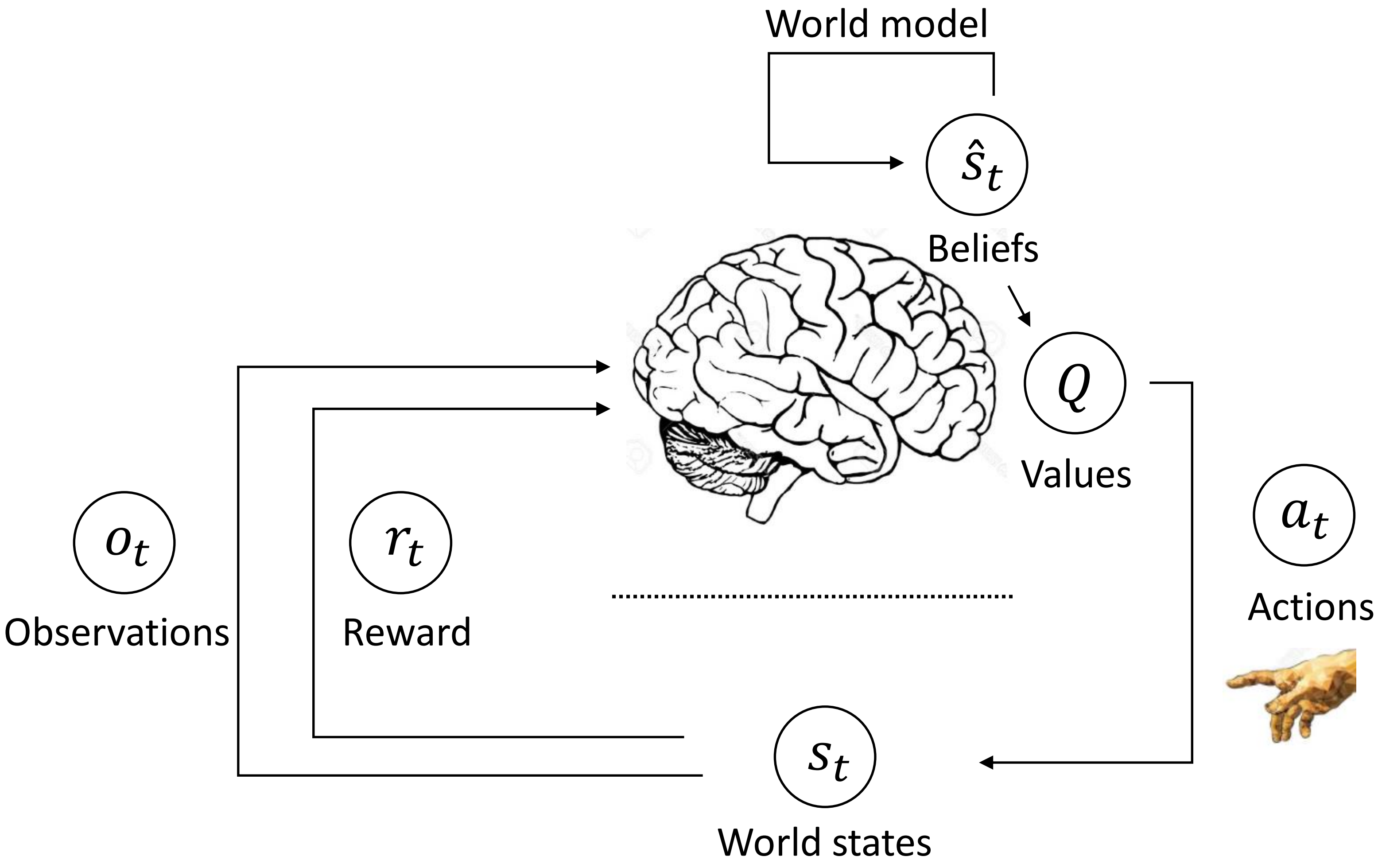


So far, we have neglected a central aspect: **state uncertainty**!

- Can we integrate beliefs about states with the MDP framework? YES!! => POMDPs
- Connection to Bayesian approaches

III. Belief-based MDPs (POMDPs)

Partially Observable Markov Decision Processes

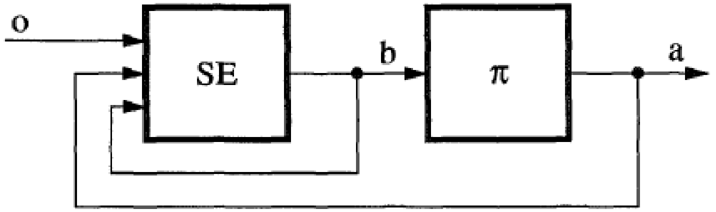


Partially Observable Markov Decision Processes

Another way to think about this is in terms of a distinction between a ‘**state estimator**’ and a ‘**policy unit**’:

State estimation **Output**:
- Belief state

State estimation **Input**:
- Observation
- Action
- Belief state



Cassandra, Kaelbling & Littman, 1994

Policy unit **Output**:
- Action

Policy unit **Input**:
- Belief state

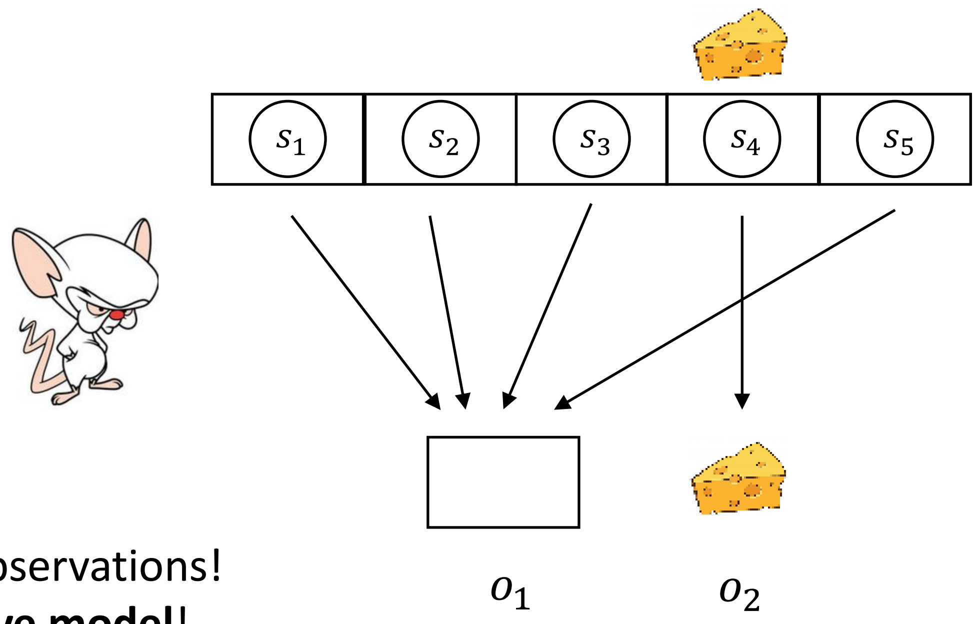
Think of a POMDP like an MDP with belief states

Couldn’t we just treat observations as states??

No! Various ways in which this could **violate** the **Markov Property**
- *Different observations* may require the *same action* because they belong to the *same (hidden) state*!
- The *same observations* may require *different actions* because they belong to *different (hidden) states*!

Partially Observable Markov Decision Processes

Let's assume **states are not directly observable** anymore – this requires agents to be even cleverer than before:



Solution: **infer belief states** based on observations!

- How? Build and invert **generative model**!

$$posterior \propto likelihood \times prior$$

$$b(s') \propto P(o|s',a,b)P(s'|a,b)$$

$$= P(o|s',a,b) \sum_{s} P(s'|s,a)\, b(s)$$

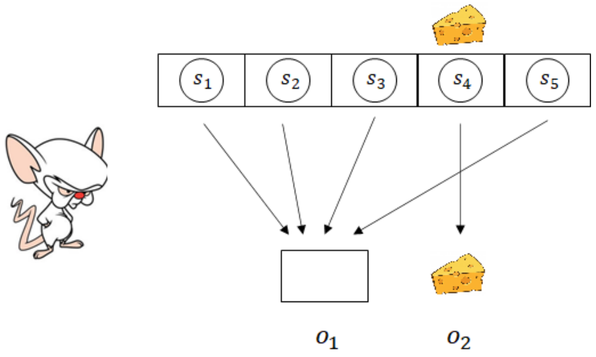
How **likely is an observation** under a hidden state, your action and belief structure

What do you **expect about the next state** given your action and (beliefs about) the previous state?

Partially Observable Markov Decision Processes

We add observations and an observation model to define a **Partially Observable Markov Decision Process**

- A (finite) state space S
- Finite set of actions A
- Finite set of **observations O**
- Transition probabilities $P(\mathrm{S_{t+1}} = \mathrm{s'} | \mathrm{S_t} = \mathrm{s}, \mathrm{A_t} = \mathrm{a})$
- Reward function $R_s = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$
- **Observation model**
- Discount factor $\gamma \in [0,1]$



Transition probabilities

to

e.g. $$P(s'|s,right) = \begin{array}{cc} & \begin{matrix} s_1 & s_2 & s_3 & s_4 & s_5 & \text{sleep} \end{matrix} \\ \begin{matrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ \end{matrix} & \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$ from

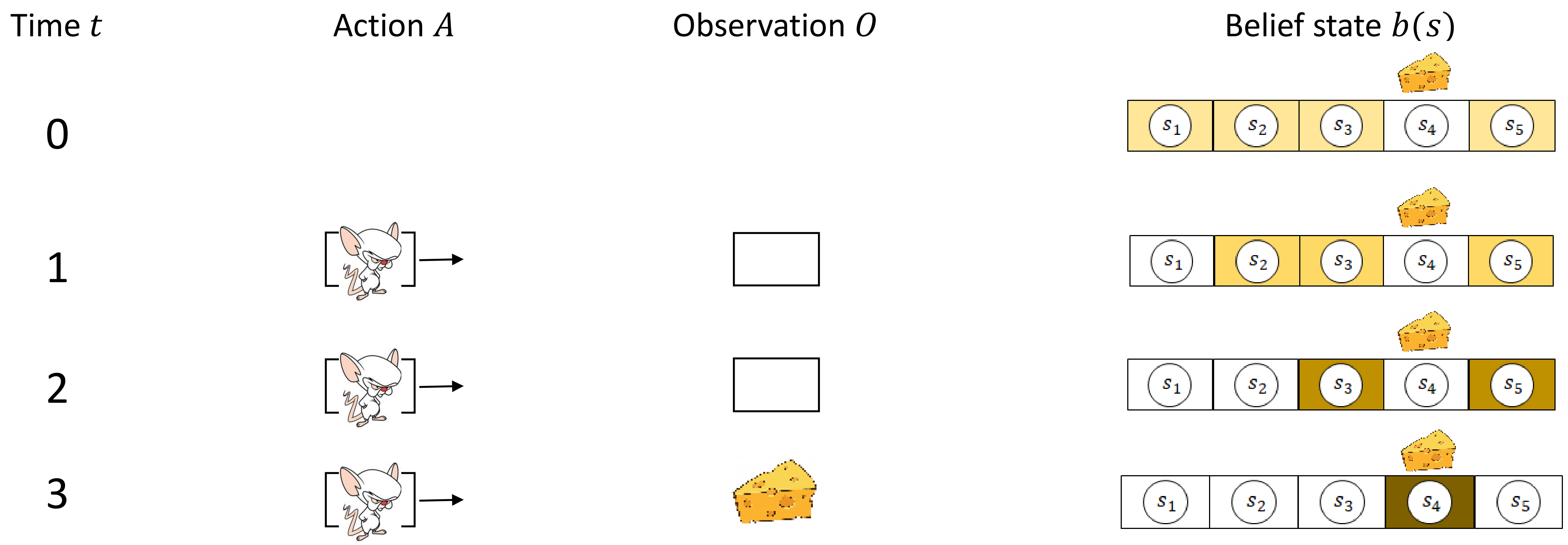
Observation model

to

$$P(o|s,right) = \begin{array}{cc} \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} & \begin{matrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{matrix} \end{array}$$ from

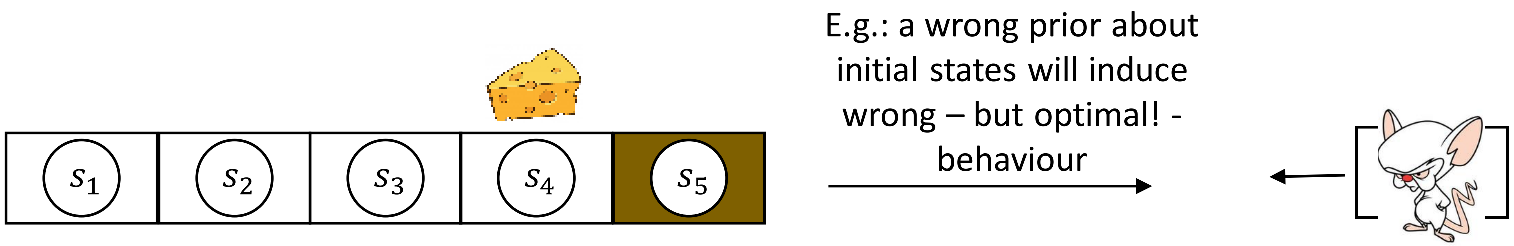
Environment dynamics specify a **Hidden Markov Model** (Markov Process + observation model)

The formation of belief states



Important link to psychiatry: suboptimal beliefs can induce wrong behaviour that is optimal wrt beliefs!

- **'Rationalisable Irrationalities'** *(e.g. Huys, Masip, Dolan & Dayan, 2015; Dayan, 2014; Schwartenbeck et al., 2015)*



Solving POMDPs

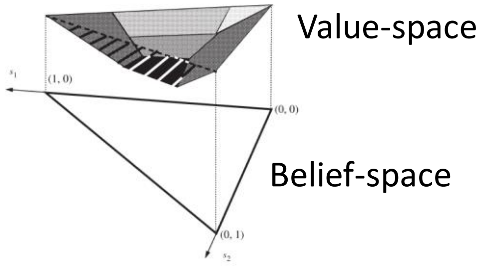
Construct reward function ρ by taking expectations over R based on belief state:

$$\rho(b,a) = \sum_{s} b(s)R(s,a)$$

Now optimise:

$$\pi_*(a|s) = \underset{\pi}{\operatorname{argmax}} \sum_{k=0}^{\infty} \gamma^k \rho(b,a)$$

For details see Kaelbling, Littman, Cassandra: *Planning and acting in partially observable stochastic domains*. Artificial Intelligence, 1998



Partially observable problems can be converted into MDPs!

The value of information

aka should you watch a talk at CPC or go for a swim with your friends?
[you should watch the talk, it will be interesting!]

Let's solve the problem using POMDPs:

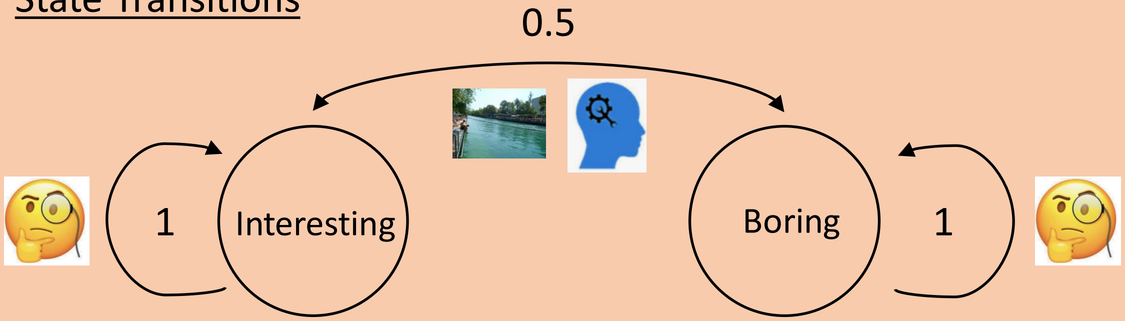
<u>Rewards</u>

		State	
		Talk interesting	Talk boring
Action	Attend talk	+10	-100
	Go swimming	-100	+10
	Check	-1	-1

<u>'Checking validity'</u>

		State	
		Talk interesting	Talk boring
Observation	 This looks interesting	0.85	0.15
	This looks boring	0.15	0.85

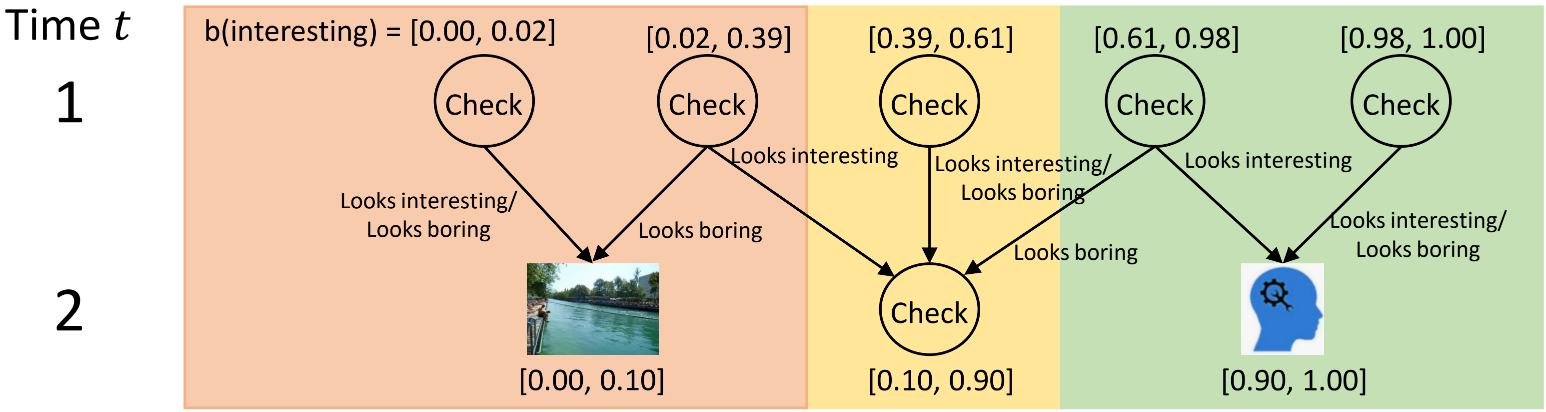
<u>State Transitions</u>



This is actually the '*Tiger problem*' (Kaelbling, Littman & Cassandra, 1998)

See also Rigoux CPC slides 2015:
https://www.tnu.ethz.ch/de/teaching/past-semesters/hs2016/cpcourse2016

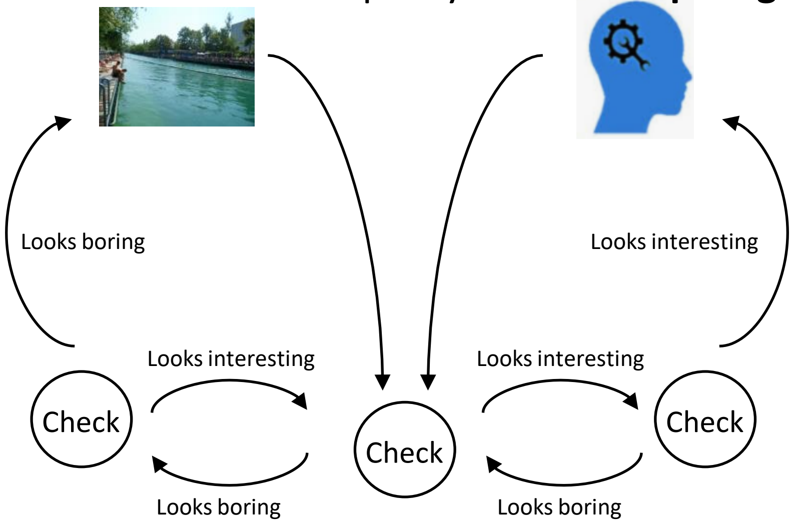
The value of information



Five different ‘policy trees’

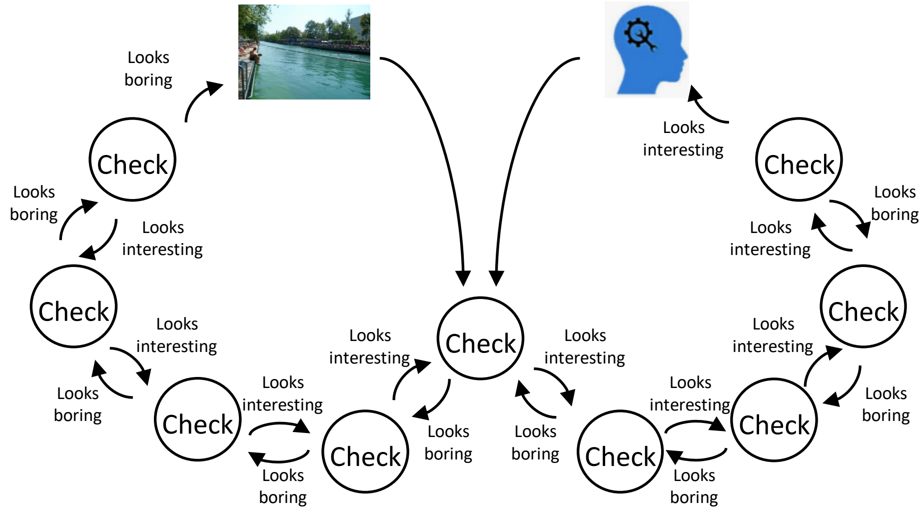
See Kaelbling, Littman & Cassandra, 1998

Optimal infinite-horizon policy drawn as a **plan graph**:



This strongly depends on the parameterisation!

Plan graph for **checking reliability reduced** to 0.65:



This relates to the **value of information** (Shannon, 1950; Howard, 1966)

- Incur a cost to move from a high entropy to a low entropy belief state – if information is higher than cost!
- Unified treatment for actions about reward (change of world state) and information gain

POMPDs Summary

Belief-based MDPs (‘POMDPs’) introduce **Bayesian inference** into decision-making/RL

- Allow to model state uncertainty
- Divides the problem into ‘**state estimation**’ and ‘**policy selection**’

State estimation: infer a belief state based on a **generative model** of the world

- Based on observations, actions and previous belief states

Policy selection: **sample action** based on belief state

Introduces at least two additional concepts that are highly relevant for computational psychiatry

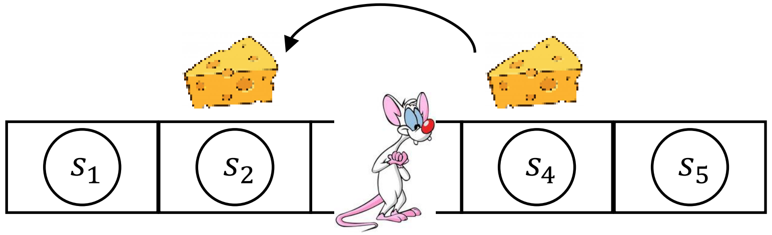
- Optimal inference/learning with **suboptimal models**
- **Value of information** – decrease uncertainty of belief states

IV. Current Research and Summary

Current Research

Computational aspects

- Multiply states by transition matrix to look into the future – complicated!
- Simplify by taking the eigenspectrum of those transitions '**intuitive planning**' *(Baram, Muller, Whittington & Behrens, 2018)*
- **Successor Representation**: decouple state and reward predictions *(Dayan, 1993; Stachenfeld, Botvinick, Gershman, 2017; Momennejad, …, Gershman, 2017)*
  - More efficient e.g. when reward function changes



- **Linear MDPs** (Piray & Daw, 2019; Todorov, 2009): linearise optimal control
  - Define objective function that includes state cost and cost of control
  - Use objective function to define a desirability function that can be solved linearly

Current Research

Abstraction and generalisation

- Mostly **toy examples** – how can we apply such frameworks to more naturalistic examples?
  - This is difficult, e.g. exploding state/action spaces
  - Summarising observations as belief states provides some generalisation
- How can biological/artificial agents **generalise knowledge** within and between tasks?
  - Lots of current research interest (e.g. Lehnert, Littman, Frank, 2020; Kemp & Tenenbaum, 2008; Wu, Schulz & Gershman, 2020; Wang, … Botvinick, 2016, 2018)

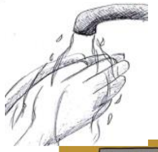
Fully Bayesian treatments, **active inference**

- Emphasis on value of information and biological implementation

Why is this relevant for computational psychiatry

Computational phenotyping – infer mechanisms that underlie suboptimal behaviour

- Suboptimal (learning of) **preferences/reward function**
- Failures of **inference on hidden states**
- Suboptimal **model configuration**
    - Action/state spaces
    - Wrong/noisy observation model, transition probabilities
- Failures of **goal-directedness/precision**
- Failures of information **gain**



Powerful framework for investigating neural mechanisms of these processes

Further Reading

Formal framework

- **Early MDP formulation**:
  - Howard: *Dynamic Programming and Markov Processes*. MIT Press, Cambridge, MA, 1960
- **MDP and RL**:
  - Sutton & Barto: *Reinforcement Learning*. MIT Press, Cambridge, MA, 1998/2018 (esp. chapter II)
- **POMDPs**
  - Kaelbling, Littman, Cassandra: *Planning and acting in partially observable stochastic domains*. Artificial Intelligence, 1998
  - Cassandra, Kaelbling, Littman: *Acting Optimally in Partially Observable Stochastic Domains*. AAAI-94 Proceedings, 1994

Gentle introduction

- Tutorial on Neuromatch Comp Neuro Summer School: https://github.com/NeuromatchAcademy/course-content/tree/master/tutorials#w2d4---optimal-control
- RL lecture David Silver at UCL: https://www.youtube.com/watch?v=lfHX2hHRMVQ
- Previous CPC material (Petzschner & Rigoux): e.g. https://bitbucket.org/fpetzschner/cpc2017/src/master/slides/

Take home messages

Markov Decision Processes provide a rich framework for modelling choice behaviour and **optimal control**

- **Markov property**
- Full specification of **dynamics of the environment** (model-based decision-making)

Various solutions of MDPs in the context of **maximising the expected return**

- **Bellman optimality**

POMDPs introduce the concept of state inference to MDPs

- **Belief-based MDPs**
- Bayesian formulations

Highly relevant for **computational psychiatry**/neuroscience

- Allow to derive '**optimal actions**', individual reward expectations, belief updating about states/actions, …
- Investigate **where things can break** and induce suboptimal behaviour/neural functioning

Thank you

Tim **Behrens**
Karl **Friston**
Ray **Dolan**

Alon **Baram**
Matt **Botvinick**
Tom **FitzGerald**
Daniel **Freinhofer**
Avital **Hahamy**
Dorothea **Hammerer**
Tobias **Hauser**
Lilla **Horvath**
Jakob **Howy**
Martin **Kronbichler**
Zeb **Kurth-Nelson**
Yunzhe **Liu**
Tamar **Makin**
Andrei **Manoliu**
Shirley **Mark**
Timothy **Müller**
Rani **Moran**
Matthew **Nour**
Thomas **Parr**
Johannes **Passecker**
Giovanni **Pezzulo**
Natalie **Rens**
Nitzan **Shahar**
Ryan **Smith**